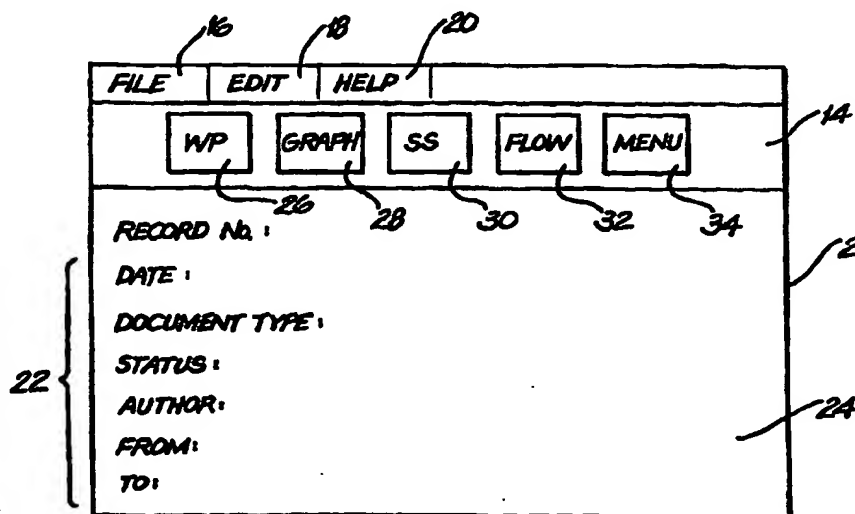




## INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification 6 : <b>G06F 9/445, 12/08, 12/02</b>		A1	(11) International Publication Number: <b>WO 95/22104</b>
			(43) International Publication Date: 17 August 1995 (17.08.95)
(21) International Application Number: PCT/AU95/00068		(81) Designated States: AM, AT, AU, BB, BG, BR, BY, CA, CH, CN, CZ, DE, DK, EE, ES, FI, GB, GE, HU, JP, KE, KG, KP, KR, KZ, LK, LR, LT, LU, LV, MD, MG, MN, MW, MX, NL, NO, NZ, PL, PT, RO, RU, SD, SE, SI, SK, TJ, TT, UA, US, UZ, VN, European patent (AT, BE, CH, DE, DK, ES, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, ML, MR, NE, SN, TD, TG), ARIPO patent (KE, MW, SD, SZ, UG).	
(22) International Filing Date: 14 February 1995 (14.02.95)			
(30) Priority Data: PM 3875 14 February 1994 (14.02.94) AU PM 8698 10 October 1994 (10.10.94) AU			
(71) Applicant (for all designated States except US): NI-TECH PTY. LIMITED [AU/AU]; Suite 1206, 88 Pitt Street, Sydney, NSW 2000 (AU).		Published With international search report.	
(72) Inventor; and (75) Inventor/Applicant (for US only): PARKER, Bruce, Peter [AU/AU]; 23 Karinya Place, Normanhurst, NSW 2076 (AU).			
(74) Agent: SPRUSON & FERGUSON; G.P.O. Box 3898, Sydney, NSW 2001 (AU).			

(54) Title: USE OF MULTIPLE APPLICATIONS AND ALLOCATION OF MEMORY OR OTHER RESOURCES IN A GUI ENVIRONMENT



## (57) Abstract

A number of GUI applications, and related data files, can be run and used at the same time. A first GUI application with a first data file or record is run. One or more primary symbols (26, 28, 30, 32) representing one or more further GUI applications are displayed. Actuation of one of the primary symbols (26, 28, 30, 32) by a user causes a search for a second data file or record associated with the first data file or record. If an associated second data file or record exists, the further GUI application runs with the associated second data file or record automatically loaded. On request for allocation of virtual memory or "Windows" resources in a Microsoft Windows environment, memory or resources greater than that requested are actually allocated.

**FOR THE PURPOSES OF INFORMATION ONLY**

Codes used to identify States party to the PCT on the front pages of pamphlets publishing international applications under the PCT.

AT	Austria	GB	United Kingdom	MR	Mauritania
AU	Australia	GE	Georgia	MW	Malawi
BB	Barbados	GN	Guinea	NE	Niger
BE	Belgium	GR	Greece	NL	Netherlands
BF	Burkina Faso	HU	Hungary	NO	Norway
BG	Bulgaria	IE	Ireland	NZ	New Zealand
BJ	Benin	IT	Italy	PL	Poland
BR	Brazil	JP	Japan	PT	Portugal
BY	Belarus	KE	Kenya	RO	Romania
CA	Canada	KG	Kyrgyzstan	RU	Russian Federation
CF	Central African Republic	KP	Democratic People's Republic of Korea	SD	Sudan
CG	Congo	KR	Republic of Korea	SE	Sweden
CH	Switzerland	KZ	Kazakhstan	SI	Slovenia
CI	Côte d'Ivoire	LJ	Liechtenstein	SK	Slovakia
CM	Cameroon	LK	Sri Lanka	SN	Senegal
CN	China	LU	Luxembourg	TD	Chad
CS	Czechoslovakia	LV	Latvia	TG	Togo
CZ	Czech Republic	MC	Monaco	TJ	Tajikistan
DE	Germany	MD	Republic of Moldova	TT	Trinidad and Tobago
DK	Denmark	MG	Madagascar	UA	Ukraine
ES	Spain	ML	Mali	US	United States of America
FI	Finland	MN	Mongolia	UZ	Uzbekistan
FR	France			VN	Viet Nam
GA	Gabon				

## USE OF MULTIPLE APPLICATIONS AND ALLOCATION OF MEMORY OR OTHER RESOURCES IN A GUI ENVIRONMENT

### Technical Field

The invention relates to a method of facilitating the use of a plurality of G.U.I. applications, and to a device for putting this method into effect. The invention is particularly useful in the field of legal work, and especially in litigation, but has uses in many other fields. A second aspect of the invention relates to a method of allocating virtual memory and resources in a G.U.I. or Windows environment.

### Background Art

The term G.U.I. is derived from the words "graphical user interface". However, for the purpose of this specification a more generalised meaning is required. Thus, in this specification, a G.U.I. application is defined to be a program for performing a particular task, such as word processing or image manipulation, in which options and commands available to the user are displayed in graphical form, and in which options and commands are represented by symbols which can be actuated by the user. The words "symbol" and "actuate" are defined below.

For the purposes of this specification, a symbol includes any symbol, whether alphanumeric, pictorial, or a combination of both. For example, the so called "icons" which are commonly used in many G.U.I. applications can be regarded as symbols.

In this specification, the word "actuate" when used in relation to a symbol should be taken to include any form of actuation, including "clicking" on a symbol using a mouse, selecting a symbol using a keyboard, pressing a symbol on a touch sensitive screen, and even using a voice actuation program to actuate a symbol by speaking into a microphone.

A file is defined to be a named set of data stored in machine readable form, for example on magnetic media or in RAM, for use with a G.U.I. application.

A data file is defined to be a file containing data which has been input by a user, manually or otherwise. A data file is created and saved by a G.U.I. application once the required data has been entered by the user of the G.U.I. application. If the user then needs to view or edit the data in a particular data file, he must first instruct the G.U.I. application to load the data file. Most G.U.I. applications allow the user to allocate names to data files and records (defined below) so as to allow the user to identify and select a particular data file or record at a later date.

A record is defined to be a group of related items of data in a data file. For example, a business can decide to store the names and addresses of its customers and suppliers using a database. In this case, the names and addresses can be stored in two separate data files called, for example, CUSTOMERS and SUPPLIERS, and each individual customer or client can occupy a separate record within each file. However, in this specification the word "record" is not limited solely to databases. Also, in this

specification, when a data file has been loaded and a particular record in that data file has been selected, that record is referred to as "loaded".

It is well known that many operating systems nowadays are multi-tasking and allow a G.U.I. application to run while one or more other G.U.I. applications are running in the background. However, in this specification, the use of the word "run" in relation to a G.U.I. application should be taken to indicate that the G.U.I. application is running in the foreground, ie. as the main application, and not in the background. This does not, of course, preclude the possibility that other G.U.I. applications are running in the background. Normally, when a G.U.I. application is running in the foreground information relating to the application is displayed to the user.

As has been stated above, the invention is particularly useful in the field of litigation. When a legal practitioner is involved in litigation on behalf of a client, it is often necessary for the legal practitioner to store a whole range of different types of information relating to the case. If a computer is used to store the information then it can be necessary to make use of a range of different software applications. For example, written documents can be recorded and edited using a word processor, images can be stored and displayed using a graphics package, general information relating to the case can be stored using a database, and financial information can be recorded using a spreadsheet. Each application, of course, makes use of its own data files, and, in order to keep track of the locations of files, it is usually necessary to store the files relating to different applications in separate directories. For example, all the word processing files can be stored in a directory called WORD, all the image files can be stored in a directory called IMAGE, and so on.

Furthermore, it can be necessary to further subdivide the data files within each directory. For example, the practitioner may want to keep all files relating to a particular client together, and may therefore wish to subdivide each directory into a number of subdirectories, each relating to a separate client. Still more subdivisions can be necessary for a variety of other reasons. For example, the practitioner can be dealing with a number of separate cases relating to the same client, and can wish to form a separate subdirectory for each case. This can result in a large and complex directory tree structure.

In litigation, it often happens that two data files relating to different applications are closely related to each other. For example, a written document, stored in a word processing data file, can make reference to a picture which is stored as an image data file. In order to read the written document the practitioner must run the word processing application and then load the data file corresponding to the written document. If he then wishes to view the picture referred to by the document, he must then run the appropriate image processing application and load the image data file corresponding to the picture. In order to do this he must first identify the file name of

the image data file. A considerable amount of time can be wasted by the practitioner in finding the file name of the image data file. Furthermore, even when the file name has been found, more time can be wasted by the practitioner in locating the data file on his computer system. It will be appreciated that the problem of locating the data file on the computer system can be particularly acute in a case where the data files of the different applications available to the practitioner are spread over a complex directory tree structure.

The invention has arisen from an attempt to overcome the above difficulties, but is applicable to a wide range of fields in addition to litigation.

#### 10 Disclosure of the Invention

According to the invention there is provided a method of facilitating the use of a plurality of G.U.I. applications, the method comprising running a first G.U.I. application with a first data file or record loaded and at the same time displaying one or more primary symbols representing one or more further G.U.I. applications, and, on actuation of one of the primary symbols by a user, searching for a second data file or record associated with said first data file or record and, if such an associated second data file or record exists, causing the further G.U.I. application represented by said one of the primary symbols to be run with the associated second data file or record automatically loaded.

20 It will be appreciated that, in the example of litigation discussed above, such a method saves the practitioner both the time involved in searching for the file name of the image file associated with his word processing file, and the time involved in locating on the computer the image file to be loaded into the graphics package.

Preferably, if no such associated second data file or record exists then, on actuation of said one of the primary symbols by a user, the method includes the step of informing the user of this fact.

In such a case, the method can include the step of, on actuation of said one of the primary symbols by a user, causing the further G.U.I. application represented by that primary symbol to be run with no data file automatically loaded.

30 Clearly, the user must have some way of specifying which data files and records are to be considered as associated with each other. Conveniently, two data files or records are considered to be associated when their names are the same, or when designated parts, for example the first parts, of their names are the same.

Preferably, the method also includes, while the first G.U.I. application is running, displaying one or more secondary symbols representing said one or more further G.U.I. applications, and, on actuation of one of said secondary symbols by a user, running the G.U.I. application associated with that secondary symbol with no data file or record automatically loaded.

It will be appreciated that this feature allows the user to switch from the first G.U.I. application to one of the further G.U.I. applications regardless of whether the two applications have associated data files or records.

Preferably, the method further includes the step of, when one of the further  
5 G.U.I. applications is running, displaying a return symbol representing the first G.U.I. application, and, on actuation of the return symbol by a user, running the first G.U.I. application.

In this case, the first G.U.I. application is preferably run with a third data file or record automatically loaded if, when the return symbol is actuated, said one of the  
10 further G.U.I. applications is running with a fourth data file or record loaded and the third and fourth data files or records are associated with each other.

Clearly, the first and third data files or records can be one and the same thing, and the second and fourth data files or records can also be one and the same thing, but this need not necessarily be the case since, after running said one of the further G.U.I.  
15 applications with the second data file or record preloaded, the user can switch to a different data file or record (ie. the fourth data file or record) before returning to the first G.U.I. application by actuating the return symbol.

In a more sophisticated embodiment of the invention, if there is more than one further G.U.I. application, the method also includes, when one of the further G.U.I.  
20 applications is running with a particular data file or record loaded, displaying one or more further application symbols representing the other further applications, and, on actuation of one of the further application symbols by a user, causing the further G.U.I. application represented by said one of the further application symbols to be run with a data file or record automatically loaded which is associated with said particular data file  
25 or record.

It will be appreciated that this feature allows the user to "hop" directly between the various different further G.U.I. applications in order to view and/or edit a complete set of related data files or records.

Ideally, the user is able to specify, at any stage during the method, the default  
30 directories to be used by the G.U.I. applications (including the first G.U.I. application) when the G.U.I. applications are caused to be run by actuation of a relevant symbol.

In the example of the legal practitioner given above, this feature allows a practitioner who happened to be working on a document in a word processing application, to switch to his graphics package and create an image data file without  
35 worrying about which directory the image data file should be saved in.

As a corollary, a further feature only allows two data files or records to be associated with each other, for example by virtue of their names, when each of the data files or records is located in a respective default directory.

It will be appreciated that this feature helps to ensure that unrelated data files and records are not accidentally associated with each other.

Conveniently, the method further includes displaying a number generating symbol and, on actuation of the number generating symbol, and input of start and finish values, by the user, generating a sequence of numbers, or alphanumeric strings, between the start and finish values.

The method can also include generating a bar code corresponding to each number or string, and/or printing a label corresponding to each number or string, and/or creating a data file or record having a name corresponding to each string.

This feature has the advantage of ensuring that human error does not arise in the creation of a sequence of bar codes, labels, or data files or records.

In the example of the legal practitioner given above, this feature can be used to create labels for a series of consecutively numbered case files, together with associated bar codes and data files or records.

The invention also provides a device for carrying out the above method, comprising computing means for running the G.U.I. applications, display means for displaying said symbols and information relating to the G.U.I. applications, first input means for allowing the user to operate the G.U.I. applications, and second input means for allowing the user to actuate said symbols.

The first and second input means can, of course, be combined with each other to form a single input means, or combined with the display means, as in the case of a touch sensitive screen.

According to a further aspect of the invention, there is provided a method of allocating virtual memory in a GUI environment, said method comprising the steps of:

providing an allocation request for virtual memory;  
increasing said allocation request to define an extra virtual memory space for storing GUI objects; and

allocating an increased virtual memory space in response to said increased allocation request when said increased allocation request is less than or equal to the amount of free virtual memory available.

According to a further aspect of the invention, there is provided a method of allocating windows resources in the Microsoft Windows<sup>®</sup> environment, said method comprising the steps of:

providing an allocation request for windows resources;  
increasing said allocation request to define a buffer for storing GUI objects;  
and

allocating an increased portion of said windows resources in response to said increased allocation request when said increased allocation request is less than or equal to the amount of free windows resources available.

According to a further aspect of the invention, there is provided a method of managing windows resources in the Microsoft Windows<sup>®</sup> environment, said method comprising the steps of:

- 5 allocating an increased percentage of windows resources for an application displayed in a main window, said increased percentage of windows resources comprising a nominal portion of windows resources for said application and a buffer which are contiguously arranged in relation to one another; and
- storing windows resources of one or more child windows of said main window in said buffer.

#### 10 Brief Description of Drawings

The preferred embodiment of the present invention will now be more particularly described, by way of example only, with reference to the accompanying drawings, in which:

- 15 Fig. 1 shows the main menu of a program for facilitating the use of a number of G.U.I. applications;

Fig. 2 shows the screen layout for a database application, when run with the program;

Fig. 3 shows the screen layout for a graphics application, when run with the program;

- 20 Fig. 4 shows the program's Utilities Menu;

Fig. 5 shows the screen layout for the program's Change Paths utility;

Fig. 6 shows the screen layout for the program's Numbering utility;

Figs 7A and 7B show allocation of windows resources in Microsoft Windows;

- 25 Figs. 8A and 8B show allocation of windows resources according to an embodiment of the invention;

Fig. 9 is a flow diagram illustrating the method of allocating free windows resources according to an embodiment of the invention;

Fig. 10 is a flow diagram illustrating step 190 of Fig. 9 in detail; and

- 30 Fig. 11 is a flow diagram illustrating the method of freeing windows resources according to an embodiment of the invention.

#### Modes for Carrying Out the Invention

- The appendix contains program listings for implementing preferred embodiments. For the sake of keeping the number of pages in the specification to a manageable size, the code in the appendix has been compressed by replacing each hard
- 35 return by the symbol "^".

The preferred embodiment is implemented by means of a program, which will be referred to simply as the program, which can be run on a conventional computer (not shown) connected to a VDU having a screen 2.

When the program is started a copyright warning message (not shown) is displayed on the screen 2. Once the user has read and accepted the copyright warning message, the program's main menu is displayed on the screen 2, as shown in Fig. 1. The main menu comprises eight symbols which can be actuated by the user by means of clicking on the appropriate symbol using a mouse (not shown). The "DB" symbol 4 represents the first, or central, application, which, in the present example, is a database application. The four symbols 6, 8, 10 and 12 on the right hand side of the main menu represent four additional applications, which will be referred to simply as the further applications for consistency with the terminology used earlier in this specification. These four symbols are labelled "WP", "GRAPH", "SS" and "FLOW", and represent a word processing application, a graphics application, a spread sheet application, and a flowchart application respectively. For reasons which will become clear below, the four symbols 6, 8, 10 and 12 will be referred to as secondary symbols. Actuation by a user of one of the secondary symbols causes the further application represented by that secondary symbol to be run. Furthermore, when a further application is run by actuating a secondary symbol, the further application is run with no data file or record automatically loaded.

When the DB symbol 4 is actuated, the central application, namely the database application, is caused to be run, and the screen shown in Fig. 2 is displayed. This screen is essentially the same as the screen which would be displayed if the database application were running normally, i.e. without the use of the program of the preferred embodiment, except that five symbols are displayed in a strip 14 across the top of the screen 2. The rest of the screen 2, outside the strip 14, is laid out in the same way as when the database is run normally. For example, the normal pull-down menus, file 16, edit 18 and help 20, are available at the top of the screen 2, and the database field names 22 are displayed down the left hand side of the data area 24 of the screen 2. The user can enter appropriate data adjacent each field name 22 in order to create a database comprising a number of records. Each record has a unique record number which is displayed at the top of the data area 24, adjacent the label "Record No.". The database application allows the user to store information in a number of different records, with each record being divided into a number of different fields.

The five symbols 26, 28, 30, 32 and 34 in the strip 14 are labelled "WP", "GRAPH", "SS", "FLOW" and "MENU", and their operation will now be described. Actuation of the MENU symbol 34 simply returns the user to the main menu shown in Fig. 1. The remaining four symbols 26, 28, 30 and 32 will be referred to as primary symbols, and represent the same four further applications as the secondary symbols 6, 8, 10 and 12. The difference between the primary and secondary symbols is that actuation of a primary symbol allows the user to run the further application represented by that primary symbol with a data file automatically loaded. When a primary symbol

is actuated the program looks to see whether there is a data file for the relevant further application having a file name (excluding the file name extension, e.g. DOC, TXT, TIF) which is identical to the record number of the record which is currently displayed by the database application. If such a data file (referred to as an associated data file) exists, the further application is caused to be run with the associated data file automatically loaded. If no such associated data file exists, a message is displayed on the screen 2 informing the user of this fact. The user is then given (in a dialogue box which is not shown in the drawings) the following three choices: to return to the screen layout of Fig. 2; to run the further application with no data file automatically loaded; or to automatically create an associated data file and run the further application with the automatically created data file automatically loaded.

As an example, the user may select record no. 1234 while using the database application shown in Fig. 2, and then click on the "GRAPH" symbol 28. The program then looks, in the default directory (explained below) of the graphics application, to see whether an image data file named 1234 exists. If an image file named 1234 exists, the graphics application will be run with the image data file 1234 automatically loaded, and the screen of Fig. 3 will be displayed. The image corresponding to the image data file 1234 will be displayed in window 36, and all the normal pull-down menus 38 of the graphics application will be available to the user. It will be seen, therefore, that the user can jump straight from a database record to the image associated with that database record, displayed in the graphics application. The program also provides a RETURN symbol 40, actuation of which returns the user to the screen view of Fig. 2. If the image data file 1234 is displayed in the graphics application when the return symbol 40 is actuated, the user is returned to database record no. 1234, i.e. the original record from which the user moved to the graphics application. However, if the user switches to a different image data file before actuating the return symbol 40, the user is given the option of returning to the database record associated with that different image data file, assuming such an associated record exists, or of returning to the original database record, i.e. record no. 1234. If no such associated record exists, the user is returned to the original database record, i.e. record no. 1234. Although this example has been given in relation to the graphics application, the same process applies for the remaining further applications.

It will be seen, therefore, that if the user wishes to associate a data file with a particular database record, all that the user has to do is to save the data file with a file name which corresponds to the record number of the database record.

When the utilities symbol 42 of the main menu (shown in Fig. 1) is actuated, the utilities menu screen shown in Fig. 4 is displayed. The utilities menu screen has ten symbols which allow the user to make use of various utilities provided by the program, and described below.

Actuation of the "Change Paths" symbol 44 displays the Paths screen shown in Fig. 5. The Paths screen allows the user to enter, in the boxes 46, the default directory path to be used by each of the four further applications. The default directory paths perform two useful functions.

5 Firstly, when a further application is run by actuation of a primary or secondary symbol, the default directory for that further application is the default directory specified in the relevant box 46. Alternatively a default area of a file, such as a database file, can be specified. The user can, of course, change directory while using the further application, but if he does not change directory then the further application  
10 will save files and look for existing files only in the default directory.

Secondly, when a primary symbol is actuated, and the program is searching for an associated data file, the program will search for the associated data file only in the relevant default directory, or the associated record only in the relevant default area of the relevant file. This is particularly useful, since it allows a number of copies of the  
15 program to be run on the same computer (not necessarily simultaneously), with each copy of the program being allocated a different set of default directories. It will be seen that it is, for example, possible to have two different image data files both named 1234 residing in different directories, for use by different copies of the program.

When the "Numbering" symbol 48 on the Utilities Menu is actuated by the  
20 user, the screen layout shown in Fig. 6 is displayed on the screen 2. The numbering utility is used to generate a sequence of consecutive labels, bar codes and/or database records. The user can enter a starting number of up to seven digits in the "Starting No." box 50, and an ending number of up to seven digits in the "Ending No." box 52. When the "Generater" symbol 54 is actuated, a sequence of numbers between the  
25 starting and ending values is generated. It is also possible to give the numbers a fixed prefix by entering a letter or number in the "Prefix" box 56. In addition, a letter can be entered in the "From" box 58, and another letter entered in the "To" box 60, in order to provide each number with a sequence of extensions between the "From" and "To" letters. In this context, an extension is a letter which is placed at the end of the  
30 number. If the "Bar Codes" option 62 is selected by the user, a bar code will be generated corresponding to each of the generated numbers. If the "Database Records" option 64 is selected, then database records will be generated having record numbers corresponding to the generated numbers. The "View" symbol 66 simply allows the  
35 user to view the generated numbers on the screen 2, before database records, bar codes, or labels are generated. The "Print" symbol 68 is used to activate a print run which actually prints the labels and bar codes. The "Cancel" symbol 70 returns the user to the utilities menu of Fig. 4 without generating any labels, bar codes or database records.

The remaining options on the utilities menu of Fig. 4 will be discussed briefly below.

Actuation of the "Resource Calculator" symbol 72 displays a calculator on the screen 2 which can be used for a variety of purposes, including calculating the total amount to be charged to a client for performing a particular task.

5 The "OCR Capture" symbol 74 allows the user to scan documents into the word processing application, and perform optical character recognition on the text of those documents.

The "Convert File" symbol 76 allows the user to convert word processing documents from one format to another, and the "Join Files" symbol 78 allows the user to join a number of different word processing files together.

10 The "Audio Read Back" symbol 80 initiates any standard program for reading the text of a document aloud, and the "Voice Control" symbol 82 initiates any standard program which allows the computer to be controlled by means of the user speaking commands into a microphone (not shown).

15 The "Image Capture" symbol 84 initiates the graphics application, and allows the user to scan in new images.

The "MENU" symbol 86 returns the user to the main menu of Fig. 1.

Returning to the main menu of Fig. 1, the "Print Reports" symbol 88 is used to print a number of predefined database reports. For example, the "Print Reports" symbol 88 can be used to print out all database records between two particular dates.

20 The "Print Reports" symbol 88 can also be used to print out information relating to the program itself, for example, a list of the default directories used by the various applications.

Actuation of the "EXIT" symbol 90 causes the program to close all open data files, stop running all applications, and terminate the program itself. If the program comes across a data file of any one of the applications which has been amended and not saved, known as a "dirty" data file, then, as a safety measure, the program asks the user whether the dirty data file is to be saved before being closed. In addition, the user is also given the option of saving dirty files, if any, whenever the RETURN symbol 40 is actuated.

30 In a further, non-illustrated embodiment of the invention, the primary symbols 26, 28, 30 and 32, and the MENU symbol 34 are displayed not only in the database application as shown in Fig. 2, but also on the screen layouts of all of the further applications. In this case, the five symbols 26 to 34 operate in exactly the same way as has been described above, in order to allow the user to jump directly from a data file in one further application to the associated data file in another further application. As above, two data files are considered to be associated if they have the same file name and are both located in the default directories which are specified by the user in the screen layout of Fig. 5.

Although the invention has been described above in relation to a database application, being the central application, and a number of further applications the data files of which may be associated with records of the database, other arrangements are possible. For example, in other embodiments of the invention, data files of the central application can be associated with data files of the further applications, or with records of the further applications. Alternatively, records of the central application can be associated with records of the further applications.

It is commonly known that many G.U.I.s, such as Windows, suffer from problems known by such terms as General Protection Faults (GPFs) or Terminal Application errors, for example. One particular form of such faults is a memory access fault. In the Windows environment, memory access is controlled by the memory manager files such as KRNL386.EXE, EMM386.EXE and SMARTDRV.EXE for example.

Windows environment applications programs, when run (launched), grab a portion of available virtual memory space which can be any combination of RAM and disk space. Thus various applications occupy contiguous blocks of virtual memory. Memory GPFs occur where an applications program attempts to read or write data (i.e. such as a data file) outside of the allocated virtual memory space, and a collision with data already residing at that location occurs, else a virtual memory address cannot be found to which to write the data.

The technique adopted in the prior art that seeks to resolve memory access faults is to reserve a proportion of the total available virtual memory space and prohibit applications programs to reside in that space. As a rule of thumb, at least 30%, and often as much as 50% of the available memory space is held in reserve.

The embodiment described in the appended program listing achieves a result where near zero memory GPFs occur. The three files, Romulus.exe, Romlinkd.dll and Romulus.dll, have effect over the memory manager file (such as KRNL386.EXE in the Windows environment). During the assignment of virtual memory space these files cause extra virtual memory space to be allocated for each applications program. The extra space is for the purposes of storing (again in the Windows environment) the window name, window class, desk class (if any) and child class. A further approximate 10% margin also is included. This extra space is sufficient such that any 'overflow' data sought to be written within the nominal virtual memory space can be accommodated, thus memory GPFs rarely if ever occur.

A further advantage, therefore, is that the prior art practice of reserving a significant portion of the available virtual memory space need not be continued, thus the full available virtual memory space can be effectively utilised.

A further beneficial aspect of the preferred embodiment is that in the instance of a Windows applications program being consigned to the 'background', the invention

functions to release the virtual memory space previously occupied by the applications program, in distinction to the prior art practices of merely closing (i.e. stop the running of) the ".exe" file. This has particular advantage of avoiding the effect of 'memory leakage'. Memory leakage can be described as the phenomenon whereby if the same applications program is continually opened then closed, the total available system resources will unavoidably reduce with each opening and closing instance. This is so even though the applications program seeks only to occupy the same virtual memory space each time it is opened.

A number of operating systems and environments currently exist which utilise a G.U.I. Examples of such G.U.I. environments include Microsoft Windows®, OS/2®, X-Windows, and the Apple operating system, all of which are well known in the art.

An embodiment of the invention is utilised in the MS Windows environment as will be described below.

The MS Windows environment incorporates a memory management system in which both random access memory (RAM) and hard drive disk space can be utilised as a single, contiguous addressable memory space for storing applications and data. The usage of a portion of available hard drive disk space in this manner is commonly referred to as virtual memory. Ideally, using this arrangement of virtual memory space, a G.U.I. environment such as MS Windows should be able to store more applications and data, as well as larger size applications and data, in memory than would be the case if only RAM were utilised.

For example, the amount of free memory available immediately after loading MS Windows without using virtual memory might be 4 MB. However, by utilising a 10 MB swapfile (that is, virtual memory), the total free memory available after loading MS Windows is increased to nearly 14 MB. Thus, a total of 14 MB of addressable memory, rather than 4 MB, is available for loading and executing both applications and data.

The size of the virtual memory space can be adjusted dependent upon available free hard drive disk space, as is well known in the art. Thus, for example, a 1 GB hard drive having 800 MB of free disk space available can be utilised under the MS Windows environment to provide virtual memory up to an upper limit.

In MS Windows, each application currently loaded in memory is displayed within a window. Similarly, each instance of a data file loaded into memory for such an application is displayed within a window as well. A region of memory known as the Windows Resources is also maintained for storing information such as the name, size, position, font, etc., of each and every graphical windows object active in the MS Windows environment, as is well known in the art. The structure known as the Windows Resources of MS Windows will be denoted by capitalisation, whereas the

windows resources, which the Windows Resources is comprised of, will not be capitalised.

The total addressable memory space, which includes both RAM and virtual memory, and the Windows Resources are controlled by a memory manager such as  
5 KRNL386.EXE. The Windows Resources itself comprises one or more fixed portions of the total addressable memory space, which each are 64 KB in size, for storing windows resources.

Thus, each application and data file loaded into the MS Windows environment is stored in a region(s) of the total addressable memory space and the corresponding  
10 window, which the application or data files are displayed in, has windows resources stored in the Windows Resources. That is, in order for an application or a data file to be loaded into the MS Windows environment, both a portion of the total addressable memory and a portion of the Windows Resources must be allocated for each application or data file.

15 When the MS Windows environment is launched, it grabs a portion of the Windows Resources for its own configuration. Each application when launched also attempts to grab the largest possible slot available of Windows Resources to meet its allocation requirement.

In the MS Windows environment, the primary restriction limiting the number  
20 of applications and/or data files that can be opened and operated safely and in a stable environment is not typically limited by the size of the total addressable memory space, since this can be enlarged dependent upon the available free hard drive disk space, but by the Windows Resources as will be described below.

Using virtual memory space, it would appear that a large number of  
25 applications and data files could be maintained in memory in the MS Windows environment by utilising virtual memory. However, this in fact is not the case as the MS Windows environment becomes increasingly unstable as more and more applications and data files are loaded into memory. That is, it is commonly known that many G.U.I.'s, such as MS Windows, suffer from problems known as General  
30 Protection Faults (GPFs). General Protection Faults are produced by a memory access fault and are strongly influenced in their likelihood of occurring by the amount of free Windows Resources available rather than by the size of the total addressable memory available.

In early versions of the MS Windows environment, the total size of the  
35 Windows Resources was limited to 64 KB. The size of the Windows Resources was therefore one factor affecting the likelihood of a GPF occurring.

However, using current computer technology, this problem of the fixed size Windows Resources is partially alleviated by the usage of hardware and software memory caching and virtual memory. Computer systems using such caching allow

filled blocks or portions of the Windows Resources to be "swapped out" of memory and onto the disk space of a hard drive temporarily until the particular portion of cache memory is again required. An empty block is swapped into the Windows Resources accordingly. Nonetheless, GPFs continue to be a primary limitation of the number of applications and data files that can be loaded into memory and have the MS Windows environment operate in a stable manner.

In fact, GPFs continue to occur with little or no reduction in their rate of occurrence and are generated by improper memory accesses into the Windows Resources in which one application attempts to access a portion of the Windows Resources allocated to another application or data file, thereby generating a GPF which not only causes the currently executing application to fail but also makes the MS Windows environment unstable. A user must therefore restart the entire MS Windows environment after a GPF occurs.

Fig. 7A is a diagram illustrating the allocation of Windows Resources 100 in the MS Windows environment for several applications and data files in accordance with the prior art.

A first application (APP1) is initiated in block 105 by double-clicking on the application icon, for example, and then opened and loaded into memory as shown in block 106. A block (or portion) 120 of the Windows Resources 100 is then allocated for first application.

The block 120 is located between addresses A0 and A2 of the Windows Resources 100 for storing the main window of the first application. As is well known in the art, the amount of windows resources required by an application is typically expressed as a percentage of the Windows Resources 100, and the percentage required by any particular application is heavily dependent upon the application. Thus, for example, an application for displaying computer animations may require 25% of Windows Resources 100, whereas a word processing application may only require 15% of Windows Resources 100.

Another disadvantage of the prior art allocation of Windows Resources 100 is a phenomenon known as memory leakage in which the entire block allocated for an application is not freed when it is closed. For example, if the first application is closed, the block 120 of Windows Resources 100 between addresses A0 and A2 should be freed for other applications to use, however, a leakage memory 130 between addresses A0 and A1 in Fig. 7A (the upper address of the memory leakage block 130 as indicated by a dashed line) is not freed and therefore is not available for other applications to use. Thus, for example, an application allocated 25% of Windows Resources 100 when it is initially loaded only releases 23% of Windows Resources 100 when it closes, thereby reducing the total amount of Windows Resources available for other applications. Frequent opening and closing of applications leads to a considerable

reduction in the amount of free Windows Resources 100 available due to the existence of these leakage memory blocks (130-134).

5 A first document (DOC1) is opened by the first application shown in block 107, and a portion 121 of the Windows Resources 100 between addresses A2 and A4 is allocated for the first document. While block 121 of the Windows Resources for the first document is shown contiguous with the block 120 of Windows Resources 100 for the first application, it will be apparent to a person skilled in the art that the prior art allocation method for Windows Resources 100 does not require this to be the case. In fact, Windows Resources 100 are allocated in a manner in which the largest available slot of the free Windows Resources 100 is allocated when the application or the data file is loaded and the remainder of the request is stored in other blocks if the request is not wholly satisfied. Thus, an application requiring 25% of the Windows Resources 100 can be allocated in several non-contiguous regions of the Windows Resources 100. For example, three non-contiguous portions of the Windows Resources 100 having sizes of 15%, 6%, and 4%, respectively, can be allocated for the application.

15 When the block 121 of the Windows Resources 100 for the first document is freed by closing the window containing the first document, this block is also subject to memory leakage as shown by the memory leakage block 131 between addresses A2 and A3.

20 A second application (APP2) is initiated as shown in the block 113, and is opened as shown in block 114. A portion 125 of the Windows Resources 100 is allocated between memory addresses A4 and A6 for the second application. Again, it will be apparent to a person skilled in the art that, while the allocation 125 of Windows Resources 100 for the second application is shown as a single contiguous block to simplify the diagram, the allocation 125 could consist of several smaller, non-contiguous portions of the Windows Resources 100.

The first application opened in block 106 opens a second data file (DOC2) in block 108, such as a document for a word processor, and is allocated a portion 122 the free Windows Resources 100 located between addresses A6 and A8.

30 A first chart (CHART1) opened by the second application shown in block 115 is allocated a portion 126 of the Windows Resources 100 between addresses A8 and A10. Both portions 122 and 126 are subject to having a leakage memory 133 and 134, respectively.

35 A region 127 of free Windows Resources 100 (indicated by hatching) remains available between addresses A10 and A11 as shown in Fig. 7A.

It will be apparent to a person skilled in the art that Fig. 7A does not necessarily represent a chronological allocation of memory since a window or child window is allocated on the basis of the largest slot of Windows Resources 100 available when an application or a data file is opened. Allocation of the percentage of Windows

Resources 100 requested by an application or file is allocated on basis of the largest slots available. Thus, the windows resources for a window or a child window are typically allocated throughout the Windows Resources 100 in non-contiguous blocks. That is, the requested windows resources are allocated by "hopping" through the  
5 Windows Resources and grabbing the largest blocks available. This method of allocating Windows Resources 100 leads to, and results in, general protection faults occurring by its very nature. Thus, while 60% of Windows Resources 100 might be free (that is, only 40% of the Windows Resources 100 are used), the MS Windows environment becomes increasingly unstable as more windows for applications and data  
10 files are opened and is prone to general protection faults occurring as the free Windows Resources 100 decreases.

Many general protection faults occur because an application attempts to overwrite the Windows Resources 100 allocated for another application. This is illustrated in Fig. 7B in which an application APP3 is allocated a block 137 of  
15 Windows Resources 100 up to address A12 and another application APP4 is allocated block 138 which starts at address A12. A general protection fault will occur in the MS Windows environment when the application APP3 attempts to increase its allocation of Windows Resources 100 by writing into the block 139 (indicated by hatching) between addresses A12 and A13, which was allocated to the application APP4 as part of its  
20 block 138. The overwriting of the windows resources of the application APP4 by the application APP3 will produce a GPF.

Thus, the method of allocating Windows Resources 100 by "hopping around" to allocate resources is a severe limitation affecting the total number of windows, and therefore the number of applications and data files, that can be opened at any given  
25 time. Many users of the MS Windows environment therefore find it necessary to close down applications to free up Windows Resources 100 in order to load other applications and data files. Thus, the total available addressable memory space comprising RAM and virtual memory does not limit the number of applications that can be loaded, but instead it is the method of allocating Windows Resources that does this.

A user is generally alerted to a general protection fault by a dialog box appearing on the screen. The user is then required to either click an OK button to exit out of the application or click an IGNORE button to continue. Regardless of the choice, many applications frequently fail at this point. The general protection fault effectively kills the application which is currently in the foreground of the MS  
30 Windows environment, although it may not have been the application that generated the GPF. While no longer useable, the .DLL module of the application remains in memory and the allocated portion of the Windows Resources 100 is not freed. This is another problem of the prior art which is distinct from the problem of memory leakage. Subsequent execution of the application generally fails since the .DLL module of the

application already exists in memory, which is detected by the application, but which cannot be communicated with.

Thus, the memory manager of the MS Windows environment has a number of longstanding disadvantages. The method of allocating Windows Resources 100 on the largest-slot-available basis and "hopping around" as each and every window is opened leads to fragmented allocation of Windows Resources 100. In this connection, child windows of an application are allocated according to this method, leading to further fragmentation of Windows Resources and consequently to general protection faults. Thus, the prior art method of allocating Windows Resources 100 limits the number of applications and data files that can be opened. Furthermore, the problem of memory leakage is exacerbated by the opening and closing of windows allocated in this manner in response to users attempting to keep the MS Windows environment stable by freeing Windows Resources 100 so that other applications to be loaded.

Thus, the most serious disadvantage affecting the number of applications and data files that can be opened in windows into the MS Windows environment is not dependent on the total addressable memory space or the size of the Windows Resources, since the Windows Resources can be swapped out using caching hardware and software, but by the prior art method of allocating Windows Resources 100 which produces memory access faults and other GPFs. The effect of using virtual memory in the MS Windows environment is that it effectively increases the running of applications so that they operate faster, but it does not allow more applications to be loaded.

An embodiment of the invention for allocating Windows Resources 100 to dramatically reduce the occurrences of GPFs will now be described with reference to Figs. 8A and 8B. A first application (APP1) is initiated in block 140 shown in Fig. 8A and is opened in block 141. A portion 155 of the Windows Resources 100 is allocated between addresses A0 and A2 for the first application.

The portion 155 of the Windows Resources 100 allocated for the first application APP1 includes a nominal portion 155A and a buffer 155B shown in Fig. 8B for storing child windows, thereby dramatically reducing the occurrences of GPFs and more fully utilising the entire Windows Resources 100, as described below. The buffer region 155B of the portion 155 of the Windows Resources 155 is indicated at the upper end of the block 155 by a dashed-dotted line. Subsequently opened data files shown in blocks 142 and 143 are provided Windows Resources 100 for their corresponding windows from the buffer 155B of the block 155 of the Windows Resources 100 allocated for the first application. This is in contrast to the separate, non-contiguous portions 121 and 122 of the Windows Resources 100 of the first and second data files of blocks 107 and 108 as shown in Fig. 7A. The likelihood of general protection faults is dramatically reduced since the child windows do not have blocks of windows resources allocated by "hopping" around.

Thus, by providing a buffer 155B in the allocation of Windows Resources 100 for an application to store window objects, the occurrences of GPFs can be greatly reduced and the Windows Resources can be utilised down to 0% in contrast to the prior art in which the MS Windows environment tends to become unstable when less than 40%-60% of free Windows Resources are available. In addition, only a single leakage memory 160 occurs for the portion 155 of the Windows Resources 100 shown in Fig. 8A in comparison to the three leakage memory blocks 130, 131 and 133 for the main and child windows of the first application in Fig. 7A.

A second application (APP2) is initiated as shown in block 148 in Fig. 8A and is opened in block 149. A portion 156 of the Windows Resources 100 is then allocated for the second application. A child window of the second application containing a first chart (CHART1) shown in block 150 opened by the second application is provided an allocation of Windows Resources 100 within the buffer (not shown) of the portion 156 allocated to second application. A free portion 157 (indicated by hatching) of the Windows Resources 100 remains between memory addresses A4 and A5 in Fig. 8A. It will be apparent to a person skilled in the art that, while portions 155 and 156 of the Windows Resources 100 are illustrated as memory blocks stored in contiguous memory addresses, they can in fact comprise several non-contiguous regions.

In this embodiment, the windows resources requested by each application are increased by 10% generally to provide such a buffer for storing the windows resources of the child windows. Typically, child windows require half of a percent of free Windows Resources 100. However, as mentioned above, applications require varying amounts of Windows Resources 100 and thus various size buffers (typically expressed as a percentage of the free Windows Resources) can be allocated for each main window. The buffer allocation of Windows Resources 100 for each application can be independently hard coded into the memory manager, or optionally can be independently configured using an initialisation file (.INI file) or other predetermined means for storing such data including a database record, etc., stored on a hard disk, a floppy disk, or other storage means such as a ROM, or can be a combination thereof.

Preferably, the Windows Resources is defragmented by the memory manager when the memory manager is loaded the first time. This can be done using conventional memory defragmentation techniques.

In this manner, the occurrences of general protection faults are dramatically reduced by providing buffers in the allocation of free Windows Resources 100 for each main window. Thus, when child windows are opened for data files and subsequent launchings of applications, the windows resources for the child windows are allocated in a region that is typically contiguous with the windows resources allocated for the main window.

Utilising this embodiment of the invention, it has been possible to launch 6 or 7 resource hungry applications in the MS Windows, which reduced the free Windows Resources 100 down to nearly 0%, yet the MS Windows environment operates in a stable manner. This is in marked contrast to the prior art in which the MS Windows environment becomes unstable and prone to general protection faults and other terminal application errors when four resource hungry applications are opened, and the free Windows Resources is reduced by 40%-60% (that is, 60% to 40% is still free, respectively), for example.

An embodiment of the invention is shown in Fig. 9 for allocating the Windows Resources 100 in accordance with Fig. 8A and 8B. In step 180, an instruction is received to open an application. This can be done by providing an actuation signal to open the application (for example, double-clicking on the application icon using a pointing device). In step 181, the memory manager of this embodiment is loaded into memory. In step 182, the memory manager obtains the memory parameters for the application. That is, in step 182, the percentage of free Windows Resources required by the particular application is determined by the memory manager. This value expressed as a percentage of the free Windows Resources can be stored in a data file.

In step 183, the memory manager finds or otherwise locates the application. This is done by identifying the window handle of the application. In decision block 184, a check is made to determine if the particular application is already loaded into the total addressable memory space. When decision block 184 returns true (yes), execution continues at step 186 in which a the child window is initiated for the application. In step 187, the memory manager assigns a portion of the buffer (e.g. a portion of buffer 155B of Fig. 8B) to the child window from the allocation of Windows Resources 100 (e.g. block 155) for the application. Execution then returns to the calling procedure.

Otherwise, when decision block 184 returns false (no), execution continues at step 185 in which the parent window of the application is initiated. In step 188, the application issues a system resource request for the main window. In step 189, the memory manager intercepts the system resource request. That is, the system resource request is not delivered to the memory allocation routine of the system library of the MS Windows environment. In step 190, the memory manager provides a buffer within the portion of windows resources allocated for the application.

Thus, the first time that an application is executed, a buffer is allocated in the applications allocated portion of Windows Resources 100 to store windows resources of any child windows of the application. The child windows of the application can be generated by subsequently loading the application and/or by opening one or more data files within the application. When a child window is initiated, the memory manager intercepts the system resource request and assigns a portion of the buffer to the child

window. Thus, fragmentation of the Windows Resources 100 shown in Fig. 7A is dramatically reduced.

The substeps comprising step 190 of Fig. 9 are shown in detail in Fig. 10. In step 195, the memory manager finds the request for Windows Resources 100. In step 196, the memory manager identifies the particular application. This includes identifying the percentage of free Windows Resources required by the application. In step 197, the request for Windows Resources 100 is increased by a known factor (that is, the percentage of free Windows Resources is increased) to provide a buffer for the storage of child windows of the application.

In decision block 198, a check is made to determine if there is sufficient free space available in the Windows Resources to allocate the increased portion of Windows Resources 100 that includes the buffer for the application. When decision block 198 returns faults (no), execution continues at step 199 in which a message is displayed alerting the user that the application cannot be loaded. Otherwise, when decision block 198 returns true (yes), execution continues at step 200 in which the requested Windows Resources 100 are allocated.

As previously described above, the requested windows resources for the application are allocated in the largest contiguous block(s) of free Windows Resources 100 available. Execution then returns to the main procedure illustrated in Fig. 9.

The procedure for freeing windows resources is illustrated in Fig. 11. In step 210, the closure of a window is initiated, which can be done by clicking on an icon or button. In step 211, the memory manager intercepts the request for closing the window that is sent to the MS Windows system library. In step 212, the memory manager closes the window. This includes shutting down the application and causing the .DLL module associated with the window, if necessary, to be closed.

In decision block 213, a check is made to determine if the window to be closed is the last remaining window of the application. When decision block 213 returns false (no), execution continues at step 214 in which the portion of the buffer allocated for the storing child window is freed, however, the buffer itself remains allocated to the windows resources assigned to the application. Otherwise, when decision block returns true (yes), execution continues at step 215 in which the entire windows resources allocation for the application is freed, except for the amount of windows resources frozen in the stack (i.e. leakage memory), if any. After steps 214 or 215, execution returns to the calling procedure.

Furthermore, when an application terminates due to a general protection fault, the whole block of windows resources allocated to the application, including the child windows, is closed by the memory manager. This is in contrast to the prior art in which the .DLL module of the application stays in memory. That is, when the

application closes, the embodiment of the invention sends a kill command to the .DLL module in memory, if it still exists.

The memory manager of the present invention continuously monitors resource allocation for child windows to ensure that resources are available within the buffer of the corresponding application for the child windows.

In this embodiment, the methods illustrated in Figs. 8-11 are carried out by three modules which are described in detail in the appended source code. In particular, the Romlinkd.DLL controls the usage and closing of child windows in an application. That is, the Romlinkd.DLL module monitors the allocation and availability of free Windows Resources that each child window has in the buffer allocated for the particular application. The Romulus.DLL module controls the handling of files between applications. Finally, the Romlink.EXE program handles all applications and operations that are directed to an application. As described above the Romlink.EXE program displays a floating button either in the top of the current window or in a tool bar. Thus, the Romlink.EXE program provides the buttons for initiating and closing applications. The Romlink.EXE program is continuously monitoring when a user clicks a button, attempts to close the application, or switches amongst applications using the task list of the MS Windows environment. When any of these events occurs the Romlink.EXE program closes all child windows in the application one at a time. If the child windows are dirty, it provides a SAVE prompt to save the contents of the window. When the Romlink.EXE program ends it removes the floating Romlink button. In this embodiment, the Romulus.INI file contains the full path and file names of the applications.

The program listing contained in the appendix is the subject of copyright owned by the applicant, and may not be reproduced in any way, except for the purposes of reproduction of this specification, without the express prior written authority of the applicant.

## APPENDIX

```

ROMULUS.DLL^ ROMULUS.MAK^ # Microsoft Visual C++ generated build script - Do
not modify^ PROJ = ROMULUS^ DEBUG = 0^ PROGTYPE = 1^ CALLER =
\_romulus\romlink\romlink^ ARGS = ^ DLLS = ^ D_RCDEFINES = -d_DEBUG^
R_RCDEFINES = -dNDEBUG^ ORIGIN = MSVC^ ORIGIN_VER = 1.00^ PROJPATH =
C:\_ROMULUS\ROMDLL\^ USEMFC = 0^ CC = cl^ CPP = cl^ CXX = cl^ CCREATEPCHFLAG =
^ CPPCREATEPCHFLAG = ^ CUSEPCHFLAG = ^ CPPUSEPCHFLAG = ^ FIRSTC = ROMULUS.C ^
FIRSTCPP = ^ RC = rc^ CFLAGS_D_WDLL = /nologo /G2 /W3 /Zi /AMw /Od /D "_DEBUG"
/FR /GD /Fd"ROMULUS.PDB"^ CFLAGS_R_WDLL = /nologo /W3 /AMw /O1 /D "NDEBUG" /FR
/GD ^ LFLAGS_D_WDLL = /NOLOGO /ONERROR:NOEXE /NOD /PACKC:61440 /CO /NOE
/ALIGN:16 /MAP:FULL^ LFLAGS_R_WDLL = /NOLOGO /ONERROR:NOEXE /NOD /PACKC:61440
/NOE /ALIGN:16 /MAP:FULL^ LIBS_D_WDLL = oldnames libw commdlg shell olecli
olesvr mdllcew^ LIBS_R_WDLL = oldnames libw commdlg shell olecli olesvr
mdllcew^ RCFLAGS = /nologo^ RESFLAGS = /nologo^ RUNFLAGS = ^ DEFFILE =
ROMULUS.DEF^ OBJEXT = ^ LIBS_EXT = ^ !if "$(DEBUG)" == "1"^ CFLAGS =
$(CFLAGS_D_WDLL)^ LFLAGS = $(LFLAGS_D_WDLL)^ LIBS = $(LIBS_D_WDLL)^ MAPFILE =
nul^ RCDEFINES = $(D_RCDEFINES)^ !else^ CFLAGS = $(CFLAGS_R_WDLL)^ LFLAGS =
$(LFLAGS_R_WDLL)^ LIBS = $(LIBS_R_WDLL)^ MAPFILE = nul^ RCDEFINES =
$(R_RCDEFINES)^ !endif^ !if [if exist MSVC.BND del MSVC.BND]^ !endif^ SBRS =
ROMULUS.SBR^ all: $(PROJ).DLL $(PROJ).BSC^ ROMULUS.OBJ: ROMULUS.C
$(ROMULUS_DEP)^ $(CC) $(CFLAGS) $(CCREATEPCHFLAG) /c ROMULUS.C^ ROMULUS.RES:
ROMULUS.RC $(ROMULUS_RCDEP)^ $(RC) $(RCFLAGS) $(RCDEFINES) -r ROMULUS.RC^
$(PROJ).DLL:: ROMULUS.RES^ $(PROJ).DLL:: ROMULUS.OBJ $(OBJEXT) $(DEFFILE)^
echo >NUL @<<$(PROJ).CRF^ ROMULUS.OBJ +^ $(OBJEXT)^ $(PROJ).DLL^ $(MAPFILE)^
c:\msvc\lib\+^ c:\msvc\mfc\lib\+^ c:\vb\cdk\+^ $(LIBS)^ $(DEFFILE);^ <<<
link $(LFLAGS) @$(PROJ).CRF^ $(RC) $(RESFLAGS) ROMULUS.RES $@^ @copy
$(PROJ).CRF MSVC.BND^ implib /nowep $(PROJ).LIB $(PROJ).DLL^ $(PROJ).DLL::
ROMULUS.RES^ if not exist MSVC.BND $(RC) $(RESFLAGS) ROMULUS.RES $@^ run:
$(PROJ).DLL^ $(PROJ) $(RUNFLAGS)^ $(PROJ).BSC: $(SBRS)^ basmake @<<< /o$@
$(SBRS)^ <<< ROMULUS.C^ #include <windows.h>^ #include <string.h>^ #include
<stdio.h>^ #include <stdlib.h>^ #include <commdlg.h>^ #include <direct.h>^
#include <dlgs.h>^ #include <dos.h>^ #include "resource.h"^ typedef struct {^
LPSTR lpszClassName;^ LPSTR lpszWindow;^ HWND hWnd;^ } FUZZYPARAMS;^ typedef
struct {^ LPARAM lParam;^ WPARAM wParam;^ UINT msg;^ HWND hWnd;^ } HOOKINFO;^
#define STARTMARK_LEN 4^ #define ENDMARK_LEN 5^ #define MAX_FILENAME_LEN
128 + 1^ #define BUFFER_SIZE 10000^ #define MAX_STRING_LEN 256^ #define
MAX_ITEMS_SEL 100^ /* GLOBAL VARIABLES */^ HINSTANCE ghInst;^ OPENFILENAME

```

```

ofn;^ char JoinedFileName[MAX_FILENAME_LEN];^ char
RetTempName[MAX_FILENAME_LEN];^ char SaveFileName[MAX_FILENAME_LEN];^ char
TempString[MAX_STRING_LEN];^ char TempStorage[MAX_STRING_LEN];^ HHOOK hHook =
NULL;^ FARPROC OldHook = NULL;^ HWND ghCurrentWnd, ghRomlinkWnd, ghChildWnd;^
/* FUNCTION DECLARATIONS */^ int __export CALLBACK LibMain(HINSTANCE, WORD,
WORD, LPSTR);^ int __export CALLBACK WEP(int);^ LPSTR __export CALLBACK
RomJoinFiles(HWND);^ UINT __export CALLBACK Open_Proc(HWND, UINT, WPARAM,
LPARAM);^ BOOL __export CALLBACK RomUnJoinFiles(HWND, LPSTR);^ LPSTR __export
CALLBACK RomTempName(LPSTR, LPSTR);^ LPSTR __export CALLBACK RomSaveAs(HWND);^
void FindFiles(HWND);^ void AddFile(HWND);^ BOOL JoinIt(HWND);^ void
DisplayMessage(HWND, WORD, WORD);^ HWND __export CALLBACK
FuzzyFindWindow(LPSTR, LPSTR);^ BOOL __export CALLBACK Fuzzy_Proc(HWND,
FUZZYPARAMS FAR *);^ BOOL __export CALLBACK RomKillFile(HWND, LPSTR);^ /*
LibMain */^ int __export CALLBACK LibMain(HINSTANCE hinst, WORD wDataSeg, WORD
cbHeapSize,^ LPSTR lpszCmdLine)^ {^ ghInst = hinst;^ if(cbHeapSize != 0)
UnlockData(0);^ return 1;^ }^ int __export CALLBACK WEP(int nParameter)^ {^
return 1;^ }^ LPSTR __export CALLBACK RomJoinFiles(HWND hWordWnd)^ {^ UINT i,
cbString;^ char chReplace;^ char szFilter[256];^ /* load file filters */^ if
((cbString = LoadString(ghInst, FILTERSTRINGS,^ szFilter, sizeof(szFilter)))
!= 0) {^ chReplace = szFilter[cbString - 1];^ for (i = 0; szFilter[i] != '\0';
i++) {^ if (szFilter[i] == chReplace)^ szFilter[i] = '\0';^ }^ }^
ofn.lStructSize = sizeof(OPENFILENAME);^ ofn.hwndOwner = hWordWnd;^
ofn.hInstance = ghInst;^ ofn.lpstrFilter = szFilter;^ ofn.lpstrCustomFilter =
NULL;^ ofn.nMaxCustFilter = 0;^ ofn.nFilterIndex = 1;^ ofn.lpstrFile = NULL;
ofn.nMaxFile = 0;^ ofn.lpstrFileName = NULL;^ ofn.nMaxFileName = 0;^
ofn.lpstrInitialDir = NULL;^ ofn.lpstrTitle = "Select files to join";^
ofn.Flags = OFN_SHOWHELP | OFN_PATHMUSTEXIST | OFN_FILEMUSTEXIST |^
OFN_ALLOWMULTISELECT | OFN_HIDEREADONLY | OFN_ENABLETEMPLATE |^
OFN_ENABLEHOOK;^ ofn.nFileOffset = NULL;^ ofn.nFileExtension = NULL;^
ofn.lpstrDefExt = NULL;^ ofn.lCustData = NULL;^ ofn.lpfHook = Open_Proc;^
ofn.lpTemplateName = MAKEINTRESOURCE(OPENFILE_DB);^ if(GetOpenFileName(&ofn)
== NULL)^ JoinedFileName[0] = 0;^ return(JoinedFileName);^ }^ UINT __export
CALLBACK Open_Proc(HWND hDlg, UINT msg, WPARAM wParam,^ LPARAM lParam)^ {^ int
Selected[MAX_ITEMS_SEL];^ int NumSel, i;^ char Buffer[128];^ switch(msg) {^
case WM_INITDIALOG:^ return TRUE;^ case WM_COMMAND:^ switch(wParam) {^ case
ID_FILELIST:^ if(HIWORD(lParam) == LEN_DBLCLK) {^ AddFile(hDlg);^
return TRUE;^ }^ break;^ case OPENBUTID_ADD:^ AddFile(hDlg);^ return
TRUE;^ case OPENBUTID_REMOVE:^ NumSel = (int) SendDlgItemMessage(hDlg,

```

- 24 -

```

OPENCNTID_JOINLIST, ^ LB_GETSELITEMS, MAX_ITEMS_SEL, ^ (LPARAM) (int FAR *)
Selected); ^ if (NumSel == 0) DisplayMessage(hDlg, NO_FILES_TOREMOVE, ^
JOIN_APP_NAME); ^ else { ^ for (i = NumSel - 1; i >= 0; i--) { ^
SendDlgItemMessage(hDlg, OPENCNTID_JOINLIST, LB_DELETESTRING, ^ Selected[i],
0); ^ } ^ } ^ return TRUE; ^ case OPENBUTID_JOIN: ^ if (JoinIt(hDlg) == TRUE) ^
PostMessage(hDlg, WM_COMMAND, IDABORT, (LONG) TRUE); ^ return TRUE; ^ case
IDOK: ^ GetDlgItemText(hDlg, ID_FILENAME, Buffer, sizeof(Buffer)); ^ for (i =
0; i < strlen(Buffer); i++) ^ if ((Buffer[i] == '*') || (Buffer[i] == '?'))
return FALSE; ^ return TRUE; ^ case pshHelp: ^ WinHelp(hDlg, "winhelp.hlp",
HELP_CONTENTS, 0); ^ return TRUE; ^ } ^ return FALSE; /* Allow standard
processing. */ ^ } ^ /* Allow standard processing. */ ^ return FALSE; ^ } ^ void
AddFile(HWND hDlg) ^ { ^ char Buffer[256]; ^ char FileNm[256]; ^ char
Directory[256]; ^ int Selected[MAX_ITEMS_SEL]; ^ int NumSel, i; ^
GetDlgItemText(hDlg, ID_DIRECTORYNAME, Directory, sizeof(Directory)); ^
if (Directory[strlen(Directory) - 1] != '\\') ^ strcat(Directory, "\\"); ^
NumSel = (int) SendDlgItemMessage(hDlg, ID_FILELIST, LB_GETSELITEMS, ^
MAX_ITEMS_SEL, (LPARAM) (int FAR *) Selected); ^ if (NumSel == 0)
DisplayMessage(hDlg, NO_FILES_TOADD, ^ JOIN_APP_NAME); ^ else { ^ for (i = 0; i
< NumSel; i++) { ^ SendDlgItemMessage(hDlg, ID_FILELIST, LB_GETTEXT, ^
Selected[i], (LPARAM) (LPSTR) FileNm); ^ strcpy(Buffer, Directory); ^
strcat(Buffer, FileNm); ^ AnsiLower(Buffer); ^ if (SendDlgItemMessage(hDlg,
OPENCNTID_JOINLIST, LB_FINDSTRINGEXACT, ^ (LPARAM) -1, (LPARAM) (LPSTR)
Buffer) == LB_ERR) ^ SendDlgItemMessage(hDlg, OPENCNTID_JOINLIST,
LB_ADDSTRING, ^ 0, (LPARAM) (LPSTR) Buffer); ^ } ^ } ^ BOOL JoinIt(HWND
hDlg) ^ { ^ HGLOBAL hBuff; ^ LPSTR Buffer; ^ HFILE InFile, OutFile; ^ int NumSel,
i; ^ unsigned readin; ^ char FileName[128]; ^ LPSTR TempName; ^ NumSel = (int)
SendDlgItemMessage(hDlg, OPENCNTID_JOINLIST, LB_GETCOUNT, ^ 0, 0); ^ if (NumSel
< 2) { ^ DisplayMessage(hDlg, AT_LEAST_TWO, JOIN_APP_NAME); ^ return FALSE; ^ } ^
if ((hBuff = GlobalAlloc(GPTR, BUFFER_SIZE)) == NULL) || ^ ((Buffer =
GlobalLock(hBuff)) == NULL) { ^ DisplayMessage(hDlg, NO_BUFFER,
JOIN_APP_NAME); ^ return FALSE; ^ } ^ /*TempPath = getenv("TEMP"); ^ if (TempPath
== NULL) ^ TempPath = _getcwd(NULL, NULL); ^ TempName = _tempnam(TempPath,
"JOIN"); ^ strcat(TempName, ".TXT"); ^ /*TempName = RomTempName("JOIN",
"TXT"); ^ if ((OutFile = _lcreat(TempName, 0)) == HFILE_ERROR) { ^
DisplayMessage(hDlg, NO_TEMP_JOIN, JOIN_APP_NAME); ^ GlobalUnlock(hBuff); ^
GlobalFree(hBuff); ^ return FALSE; ^ } ^ for (i = 0; i < NumSel; i++) { ^
SendDlgItemMessage(hDlg, OPENCNTID_JOINLIST, LB_GETTEXT, ^ i, (LPARAM)
(LPSTR) FileName); ^ if ((InFile = _lopen(FileName, READ)) == HFILE_ERROR) { ^

```

```

LoadString(ghInst, NO_OPEN_FILE, Buffer, MAX_STRING_LEN);^ lstrcat(Buffer,
FileName);^ /* using FileName to store the dialog title */^ LoadString(ghInst,
JOIN_APP_NAME, FileName, MAX_STRING_LEN);^ MessageBox(hDlg, Buffer, FileName,
MB_OK | MB_ICONEXCLAMATION);^ GlobalUnlock(hBuff);^ GlobalFree(hBuff);^ return
FALSE;^ }^ _lwrite(OutFile, "[***", 4);^ _lwrite(OutFile, FileName,
lstrlen(FileName));^ _lwrite(OutFile, "***\n", 5);^ while(TRUE) {^ readin =
_lread(InFile, Buffer, BUFFER_SIZE);^ if(readin != 0) _lwrite(OutFile, Buffer,
readin);^ if(readin < BUFFER_SIZE) break;^ }^ _lclose(InFile);^ }^
_lwrite(OutFile, "[***END***\n", 12);^ _lclose(OutFile);^ /* free up the
buffer */^ GlobalUnlock(hBuff);^ GlobalFree(hBuff);^ lstrcpy(JoinedFileName,
TempName);^ return TRUE;^ }^ BOOL __export CALLBACK RomUnJoinFiles(HWND
hWordWnd, LPSTR JoinedFileName)^ {^ HGLOBAL hBuff;^ LPSTR Buffer;^ HFILE
InFile, OutFile;^ unsigned readin, i, MarkerPos, StartPos, FileNamePos;^ char
DestFileName[MAX_FILENAME_LEN + ENDMARK_LEN];^ char BeginMark[STARTMARK_LEN +
1] = "[***";^ BOOL InFileName;^ if(((hBuff = GlobalAlloc(GPTR, BUFFER_SIZE))
== NULL) || ((Buffer = GlobalLock(hBuff)) == NULL)) {^
DisplayMessage(hWordWnd, NO_BUFFER, JOIN_APP_NAME);^ return FALSE;^ }^
if((InFile = _lopen(JoinedFileName, READ)) == HFILE_ERROR) {^
DisplayMessage(hWordWnd, NO_OPEN_JOIN, JOIN_APP_NAME);^ GlobalUnlock(hBuff);^
GlobalFree(hBuff);^ return FALSE;^ }^ MarkerPos = 0;^ InFileName = FALSE;^
OutFile = HFILE_ERROR;^ while(TRUE) {^ readin = _lread(InFile, Buffer,
BUFFER_SIZE);^ for(i = 0; i < readin; i++) {^ if(InFileName) {^ if(Buffer[i]
== 0x0a) {^ DestFileName[FileNamePos - ENDMARK_LEN + 1] = 0;^
if(lstrcmp(DestFileName, "END") == 0) {^ OutFile = HFILE_ERROR;^ readin
= 0;^ break;^ }^ else {^ if((OutFile = _lcreat(DestFileName, 0)) ==
HFILE_ERROR) {^ LoadString(ghInst, NO_OPEN_FILE, Buffer, MAX_STRING_LEN);^
lstrcat(Buffer, DestFileName);^ LoadString(ghInst, JOIN_APP_NAME,
DestFileName,^ MAX_STRING_LEN);^ MessageBox(hWordWnd, Buffer,
DestFileName,^ MB_OK | MB_ICONEXCLAMATION);^ GlobalUnlock(hBuff);^
GlobalFree(hBuff);^ return FALSE;^ }^ StartPos = i + 1;^ InFileName =
FALSE;^ }^ }^ else {^ DestFileName[FileNamePos] = Buffer[i];^
FileNamePos++;^ if(FileNamePos >= sizeof(DestFileName))^ FileNamePos =
sizeof(DestFileName) - 1;^ }^ }^ else {^ if(Buffer[i] ==
BeginMark[MarkerPos]) {^ if((MarkerPos == 0) && ((i + STARTMARK_LEN) >
BUFFER_SIZE)) {^ _llseek(InFile, (i - BUFFER_SIZE), 1);^ break;^ }^
MarkerPos++;^ }^ else MarkerPos = 0;^ if(MarkerPos == STARTMARK_LEN) {^
MarkerPos = 0;^ InFileName = TRUE;^ FileNamePos = 0;^ if((OutFile !=
HFILE_ERROR) && ((i - STARTMARK_LEN) > StartPos))^ _lwrite(OutFile,

```

- 26 -

```

&Buffer[StartPos], ^ i - STARTMARK_LEN - StartPos + 1); ^ _lclose(OutFile); ^
} ^ } ^ if ((!InFileName) && (OutFile != HFILE_ERROR) && (i > StartPos)) ^
_lwrite(OutFile, &Buffer[StartPos], i - StartPos); ^ StartPos = 0; ^ if(readin <
BUFFER_SIZE) break; ^ } ^ _lclose(OutFile); ^ _lclose(InFile); ^
GlobalUnlock(hBuff); ^ GlobalFree(hBuff); ^ return TRUE; ^ } ^ LPSTR __export
CALLBACK RomSaveAs(HWND hWordWnd) ^ { ^ UINT i, cbString; ^ char chReplace; ^ char
szFilter[256]; ^ if ((cbString = LoadString(ghInst, FILTERSTRINGS, ^ szFilter,
sizeof(szFilter))) != 0) { ^ chReplace = szFilter[cbString - 1]; ^ for (i = 0;
szFilter[i] != '\0'; i++) { ^ if (szFilter[i] == chReplace) ^ szFilter[i] =
'\0'; ^ } ^ } ^ ofn.lStructSize = sizeof(OPENFILENAME); ^ ofn.hwndOwner =
hWordWnd; ^ ofn.hInstance = ghInst; ^ ofn.lpstrFilter = szFilter; ^
ofn.lpstrCustomFilter = NULL; ^ ofn.nMaxCustFilter = 0; ^ ofn.nFilterIndex = 1; ^
ofn.lpstrFile = SaveFileName; ^ ofn.nMaxFile = sizeof(SaveFileName); ^
ofn.lpstrFileName = NULL; ^ ofn.nMaxFileName = 0; ^ ofn.lpstrInitialDir =
NULL; ^ ofn.lpstrTitle = "Save As"; ^ ofn.Flags = OFN_SHOWHELP |
OFN_PATHMUSTEXIST | OFN_HIDEREADONLY | ^ OFN_OVERWRITEPROMPT; ^
ofn.nFileOffset = NULL; ^ ofn.nFileExtension = NULL; ^ ofn.lpstrDefExt = NULL; ^
ofn.lCustData = NULL; ^ ofn.lpfnHook = NULL; ^ ofn.lpTemplateName = NULL; ^
if(GetSaveFileName(&ofn) == NULL) ^ SaveFileName[0] = 0; ^
return(SaveFileName); ^ } ^ void DisplayMessage(HWND hWnd, WORD wID, WORD tID) ^
{ ^ char Message[MAX_STRING_LEN]; ^ char Title[MAX_STRING_LEN]; ^
LoadString(ghInst, wID, Message, MAX_STRING_LEN); ^ LoadString(ghInst, tID,
Title, MAX_STRING_LEN); ^ MessageBox(hWnd, Message, Title, MB_OK |
MB_ICONEXCLAMATION); ^ } ^ HWND __export CALLBACK FuzzyFindWindow(LPSTR
lpzClassName, LPSTR lpzWindow) ^ { ^ FUZZYPARAMS FuzzyParams; ^
FuzzyParams.lpszClassName = (lpzClassName[0] == 0)? NULL : lpzClassName; ^
FuzzyParams.lpszWindow = (lpzWindow[0] == 0)? NULL : lpzWindow; ^
FuzzyParams.hWnd = NULL; ^ EnumWindows(Fuzzy_Proc, (DWORD) ((FUZZYPARAMS FAR
*)&FuzzyParams)); ^ return FuzzyParams.hWnd; ^ } ^ BOOL __export CALLBACK
Fuzzy_Proc(HWND hWnd, FUZZYPARAMS FAR *lpFuzzyParams) ^ { ^ char
Buffer[MAX_STRING_LEN]; ^ if(lpFuzzyParams->lpszClassName != NULL) { ^
GetClassName(hWnd, Buffer, MAX_STRING_LEN); ^ if(lstrcmp(Buffer, lpFuzzyParams-
>lpszClassName) != 0) ^ return (1); ^ } ^ if(lpFuzzyParams->lpszWindow != NULL)
{ ^ GetWindowText(hWnd, Buffer, sizeof(Buffer)); ^ if(_fstrchr(Buffer,
lpFuzzyParams->lpszWindow) == NULL) ^ return (1); ^ } ^ lpFuzzyParams->hWnd =
hWnd; ^ return(0); ^ } ^ /* BOOL __export CALLBACK RomKillFile(HWND hWnd,
LPSTR FileName) ^ { ^ lstrcpy(TempStorage, FileName); ^ while(IsWindow(hWnd)) ^
Yield(); ^ if(remove(TempStorage) == 0) ^ return TRUE; ^ return FALSE; ^ } */

```

- 27 -

```

LPSTR __export CALLBACK RomTempName(LPSTR Prefix, LPSTR Ext)^ {^ char
*TempPath, TempPrefix[6];^ int Num, i, NumPos;^ TempPath = getenv("TEMP");^
if(TempPath == NULL)^ TempPath = getcwd(NULL, NULL);^ lstrcpyn(TempPrefix,
Prefix, 5);^ lstrcpy(TempString, TempPath);^ if(TempString[lstrlen(TempString)
- 1] != '\\')^ lstrcat(TempString, "\\");^ lstrcat(TempString, TempPrefix);^
NumPos = lstrlen(TempString);^ lstrcat(TempString, "0000.");^
lstrcat(TempString, Ext);^ for(Num = 0; Num < 10000; Num++) {^
wprintf(TempStorage, "%04d", Num);^ for(i = 0; i < 4; i++)^ TempString[NumPos
+ i] = TempStorage[i];^ if(_dos_findfirst(TempString, _A_ARCH | _A_HIDDEN |
_A_NORMAL | _A_RDONLY | _A_SUBDIR | _A_SYSTEM | _A_VALID, NULL) != 0)
break;^ }^ if(Num == 10000) {^ TempString[0] = NULL;^ }^ return TempString;^
}^ ROMULUS.DEF^ LIBRARY Romulus^ DESCRIPTION 'Romulus DLL - extra
functionality for Romulus'^ EXETYPE WINDOWS^ STUB 'WINSTUB.EXE'^ CODE PRELOAD
MOVEABLE DISCARDABLE^ DATA PRELOAD MOVEABLE SINGLE^ HEAPSIZE 4096^ EXPORTS^
WEP      @1 RESIDENTNAME^ RomJoinFiles @2^ Open_Proc @3^
RomUnJoinFiles @4^ RomTempName @5^ RomSaveAs @6^ FuzzyFindWindow
@7^ Fuzzy_Proc @8^ RomKillFile @9^ JOINRC.H^ #define OPENFILE_DB
10^ #define FILTERSTRINGS 100^ #define JOIN_APP_NAME 110^ #define
CONV_APP_NAME 111^ #define JOIN_TITLE 112^ #define SAVEAS_TITLE
113^ #define NO_FILES_TOREMOVE 120^ #define NO_FILES_TOADD 121^ #define
AT_LEAST_TWO 122^ #define NO_BUFFER 123^ #define NO_TEMP_JOIN 124^
#define NO_OPEN_JOIN 125^ #define NO_OPEN_FILE 126^ #define
TEMP_ENV_VAR 130^ #define TEMP_FILE_MASK 131^ #define START_MARK 132^
#define END_MARK 133^ #define COMPLETED_MARK 134^ #define ID_FILELIST
1120^ #define ID_DIRECTORYNAME 1088^ #define ID_FILENAME 1152^ #define
OPENCNTID_JOINLIST 100^ #define OPENBUTID_ADD 101^ #define
OPENBUTID_REMOVE 102^ #define OPENBUTID_JOIN 103^ ROMLINKD.DLL^
ROMLINKD.MAK^ # Microsoft Visual C++ generated build script - Do not modify^
PROJ = ROMLINKD^ DEBUG = 0^ PROGTYPE = 1^ CALLER = ^ ARGS = ^ DLLS = ^
D_RCDEFINES = -d_DEBUG^ R_RCDEFINES = -dNDEBUG^ ORIGIN = MSVC^ ORIGIN_VER =
1.00^ PROJPATH = C:\ROMULUS\LINKDLL\^ USEMFC = 0^ CC = cl^ CPP = cl^ CXX =
cl^ CCREATEPCHFLAG = ^ CPPCREATEPCHFLAG = ^ CUSEPCHFLAG = ^ CPPUSEPCHFLAG = ^
FIRSTC = ROMLINKD.C ^ FIRSTCPP = ^ RC = rc^ CFLAGS_D_WDLL = /nologo /G2 /W3
/Zi /ALw /Od /D "_DEBUG" /FR /GD /Fd"ROMLINKD.PDB"^ CFLAGS_R_WDLL = /nologo
/W3 /AMw /O1 /D "NDEBUG" /FR /GD ^ LFLAGS_D_WDLL = /NOLOGO /ONERROR:NOEXE /NOD
/PACKC:61440 /CO /NOE /ALIGN:16 /MAP:FULL^ LFLAGS_R_WDLL = /NOLOGO
/ONERROR:NOEXE /NOD /PACKC:61440 /NOE /ALIGN:16 /MAP:FULL^ LIBS_D_WDLL =
oldnames libw commdlg shell olecli olesvr ldllcew^ LIBS_R_WDLL = oldnames libw

```

- 28 -

```

commdlg shell olecli olesvr mdllcew^ RCFLAGS = /nologo^ RESFLAGS = /nologo^
RUNFLAGS = ^ DEFFILE = ROMLINKD.DEF^ OBJEXT = ^ LIBS_EXT = ^ !if "$(DEBUG)"
== "1"^ CFLAGS = $(CFLAGS_D_WDLL)^ LFLAGS = $(LFLAGS_D_WDLL)^ LIBS =
$(LIBS_D_WDLL)^ MAPFILE = nul^ RCDEFINES = $(D_RCDEFINES)^ !else^ CFLAGS =
$(CFLAGS_R_WDLL)^ LFLAGS = $(LFLAGS_R_WDLL)^ LIBS = $(LIBS_R_WDLL)^ MAPFILE =
nul^ RCDEFINES = $(R_RCDEFINES)^ !endif^ !if [if exist MSVC.BND del MSVC.BND]^
!endif^ SERS = ROMLINKD.SBR^ ROMLINKD_DEP = ^ all: $(PROJ).DLL $(PROJ).BSC^
ROMLINKD.OBJ: ROMLINKD.C $(ROMLINKD_DEP)^ $(CC) $(CFLAGS) $(CCREATEPCHFLAG)
/c ROMLINKD.C^ $(PROJ).DLL:: ROMLINKD.OBJ $(OBJEXT) $(DEFFILE)^ echo >NUL
@<<$(PROJ).CRF^ ROMLINKD.OBJ +^ $(OBJEXT)^ $(PROJ).DLL^ $(MAPFILE)^
c:\msvc\lib\+^ c:\msvc\mfc\lib\+^ $(LIBS)^ $(DEFFILE);^ << link $(LFLAGS)
@$(PROJ).CRF^ $(RC) $(RESFLAGS) $@^ implib /nowep $(PROJ).LIB $(PROJ).DLL^
run: $(PROJ).DLL^ $(PROJ) $(RUNFLAGS)^ $(PROJ).BSC: $(SERS)^ bscmake @<<^
/o$@ $(SERS)^ << ROMLINKD.C^ #include <windows.h>^ #include <string.h>^
#include <stdio.h>^ #include <stdlib.h>^ #include <commdlg.h>^ #include
<direct.h>^ #include <dlgs.h>^ #include <dos.h>^ /* GLOBAL VARIABLES */^
HINSTANCE ghInst;^ HHOOK hHook = NULL;^ HWND ghCurrentWnd, ghRomlinkWnd,
ghChildWnd;^ HWND ghFTParent;^ HHOOK hHookCBT = NULL;^ /* FUNCTION
DECLARATIONS */^ int __export CALLBACK LibMain(HINSTANCE, WORD, WORD, LPSTR);^
int __export CALLBACK WEP(int);^ BOOL __export CALLBACK RomSet(HWND, HWND);^
BOOL __export CALLBACK RomExtraSet(HWND);^ void __export CALLBACK
RomFree(void);^ void __export CALLBACK ItFunc(int, WORD, DWORD);^ void
__export CALLBACK RomSetChildWindow(HWND);^ int __export CALLBACK UsFunc(int,
WORD, DWORD);^ /* LibMain */^ int __export CALLBACK LibMain(HINSTANCE hinst,
WORD wDataSeg, WORD cbHeapSize, LPSTR lpszCmdLine)^ {^ ghInst = hinst;^
if(cbHeapSize != 0) UnlockData(0);^ return 1;^ }^ int __export CALLBACK
WEP(int nParameter)^ {^ return 1;^ }^ BOOL __export CALLBACK RomSet(HWND hWnd,
HWND hRWnd)^ {^ ghCurrentWnd = hWnd;^ ghRomlinkWnd = hRWnd;^ ghChildWnd =
NULL;^ hHook = SetWindowsHookEx(WH_CALLWNDPROC, (HOOKPROC)(FARPROC)ItFunc,^
ghInst, NULL/*GetWindowTask(hWnd)*/);^ if(hHook) return TRUE;^ return FALSE;^
}^ BOOL __export CALLBACK RomExtraSet(HWND hWnd)^ {^ ghFTParent = hWnd;^
if((hHookCBT = SetWindowsHookEx(WH_CBT, (HOOKPROC)(FARPROC)UsFunc, ghInst,
NULL)) == NULL)^ return FALSE;^ return TRUE;^ }^ void __export CALLBACK
RomFree()^ {^ if(hHook) UnhookWindowsHookEx(hHook);^ if(hHookCBT)
UnhookWindowsHookEx(hHookCBT);^ hHook = NULL;^ }^ void __export CALLBACK
ItFunc(int nCode, WORD wParam, DWORD lParam)^ {^ /* structure used for
CallWpnProc */^ struct tempStruct {^ LONG lParam;^ WORD wParam;^ WORD
message;^ HWND hwnd;^ } FAR * lptsParamStruct;^ if( nCode >= 0 ) {^

```

- 29 -

```

lptsParamStruct = (struct tempStruct FAR *) lParam; if(lptsParamStruct-
>message == WM_NCDESTROY) { if(lptsParamStruct->hwnd == ghCurrentWnd)^
PostMessage(ghRomlinkWnd, WM_USER + 50, 0, 0); else if(lptsParamStruct->hwnd
== ghChildWnd) { PostMessage(ghRomlinkWnd, WM_USER + 51, 0, 0); ghChildWnd =
NULL; } } ^ CallNextHookEx(hHook, nCode, wParam, lParam); return; } ^
void __export CALLBACK RomSetChildWindow(HWND hWnd) { ghChildWnd = hWnd; } ^
int __export CALLBACK UsFunc(int nCode, WORD wParam, DWORD lParam) {
LPCBT_CREATEWND CreateWndParam; HWND hWndApp; if(nCode == HCBT_CREATEWND) {
CreateWndParam = (LPCBT_CREATEWND) lParam; if(CreateWndParam->lpcs-
>hwndParent == ghFTPParent) { hWndApp = GetParent(ghFTPParent);
PostMessage(ghFTPParent, WM_MDIMAXIMIZE, wParam, 0); PostMessage(hWndApp,
WM_COMMAND, 0x13b, 0); PostMessage(hWndApp, WM_COMMAND, 0x133, 0); } ^ } ^
return (int) CallNextHookEx(hHookCBT, nCode, wParam, lParam); } ^
ROMLINKD.DEF^ LIBRARY romlinkd^ DESCRIPTION 'Romulus DLL. Copyright (c) 1993 -
1994 Ni-Tech Pty Limited. Patent Pending.'^ EXETYPE WINDOWS^ STUB
'WINSTUB.EXE'^ CODE PRELOAD MOVEABLE DISCARDABLE^ DATA PRELOAD MOVEABLE
SINGLE^ HEAPSIZE 4096^ EXPORTS^ WEP @1 RESIDENTNAME^ RomSet
@2^ RomExtraSet @3^ RomFree @4^ ItFunc @5^
RomSetChildWindow @6^ UsFunc @7^ ^ ROMLINK.EXE^ ROMLINK.MAK^ #
Microsoft Visual C++ generated build script - Do not modify^ PROJ = ROMLINK^
DEBUG = 1^ PROGTYPE = 0^ CALLER = ^ ARGS = ^ DLLS = ^ D_RCDEFINES = -d_DEBUG^
R_RCDEFINES = -dNDEBUG^ ORIGIN = MSVC^ ORIGIN_VER = 1.00^ PROJPATH =
C:\A_FILES\ROMLINK\^ USEMFC = 1^ CC = cl^ CPP = cl^ CXX = cl^ CCREATEPCHFLAG =
^ CPPCREATEPCHFLAG = ^ CUSEPCHFLAG = ^ CPPUSEPCHFLAG = ^ FIRSTC = ROMLINK.C ^
FIRSTCPP = ^ RC = rc^ CFLAGS_D_WEXE = /nologo /W3 /FR /G2 /Zi /D_DEBUG /Od /AM
/GA /Fd"ROMLINK.PDB"^ CFLAGS_R_WEXE = /nologo /W3 /FR /O1 /DNDEBUG /AM /GA^
LFLAGS_D_WEXE = /NOLOGO /ONERROR:NOEXE /NOD /PACKC:61440 /CO /ALIGN:16
/STACK:10240^ LFLAGS_R_WEXE = /NOLOGO /ONERROR:NOEXE /NOD /PACKC:61440
/ALIGN:16 /STACK:10240^ LIBS_D_WEXE = mafxcwd oldnames libw commdlg shell
olecli olesvr mlibcew^ LIBS_R_WEXE = mafxcw oldnames libw commdlg shell olecli
olesvr mlibcew^ RCFLAGS = /nologo^ RESFLAGS = /nologo^ RUNFLAGS = ^ DEFFILE =
ROMLINK.DEF^ OBJEXT = ^ LIBS_EXT = ROMLINKD.LIB ^ !if "$(DEBUG)" == "1"^
CFLAGS = $(CFLAGS_D_WEXE)^ LFLAGS = $(LFLAGS_D_WEXE)^ LIBS = $(LIBS_D_WEXE)^
MAPFILE = nul^ RCDEFINES = $(D_RCDEFINES)^ !else^ CFLAGS = $(CFLAGS_R_WEXE)^
LFLAGS = $(LFLAGS_R_WEXE)^ LIBS = $(LIBS_R_WEXE)^ MAPFILE = nul^ RCDEFINES =
$(R_RCDEFINES)^ !endif^ !if [if exist MSVC.BND del MSVC.BND]^ !endif^ SERS =
ROMLINK.SER^ ROMLINK_DEP = c:\a_files\romlink\romlink.h \^
c:\a_files\romlink\romlrc.h^ ROMLINKD_DEP = ^ all: $(PROJ).EXE $(PROJ).BSC^

```

```

ROMLINK.OBJ: ROMLINK.C $(ROMLINK_DEP)^ $(CC) $(CFLAGS) $(CCREATEPCHFLAG) /c
ROMLINK.C^ $(PROJ).EXE:: ROMLINK.OBJ $(OBJS_EXT) $(DEFFILE)^ echo >NUL
@<<$(PROJ).CRF^ ROMLINK.OBJ +^ $(OBJS_EXT)^ $(PROJ).EXE^ $(MAPFILE)^
c:\msvc\lib\+^ c:\msvc\mfc\lib\+^ ROMLINKD.LIB+^ $(LIBS)^ $(DEFFILE);^ <<^
link $(LFLAGS) @$$(PROJ).CRF^ $(RC) $(RESFLAGS) $@^ run: $(PROJ).EXE^
$(PROJ) $(RUNFLAGS)^ $(PROJ).BSC: $(SBSRS)^ bscmake @<<^ /o$@ $(SBSRS)^ <<^
ROMLINK.C^ #include "Romlink.h"^ HANDLE ghInst;^ HWND ghWnd, ButWnd,
ghAppWnd;^ char AppName[200];^ char MainClass[MAX_CLASS_LEN];^ char
DeskClass[MAX_CLASS_LEN];^ char ChildClass[MAX_CLASS_LEN];^ BOOL
SpecialClose;^ BOOL HookSet = FALSE, FTHookSet = FALSE;^ int gAppID;^ BOOL
gLeaveOpen;^ int PASCAL WinMain(HANDLE hInstance, HANDLE hPrevInstance,^
LPSTR lpszCmdLine, int nCmdShow)^ {^ WNDCLASS Class;^ MSG msg;^ RECT rc; ^
if(hPrevInstance) {^ GetInstanceData(hPrevInstance, (BYTE *) &ButWnd,
sizeof(HWND));^ SetFocus(ButWnd);^ return FALSE;^ }^ ghInst = hInstance;^
LoadString(ghInst, APP_NAME, AppName, sizeof(AppName));^ Class.style = NULL;^
Class.lpfnWndProc = (WNDPROC) Main_Proc;^ Class.cbClsExtra = 0;^
Class.cbWndExtra = 0;^ Class.hInstance = ghInst;^ Class.hIcon = NULL;^
Class.hCursor = LoadCursor(NULL, IDC_ARROW);^ Class.hbrBackground = NULL;^
Class.lpszMenuName = NULL;^ Class.lpszClassName = "Romlink10";^
if(!RegisterClass(&Class)) return FALSE;^ GetWindowRect(GetDesktopWindow(),
&rc);^ ghWnd = CreateWindow(^ "Romlink10",^ "Romulus",^ WS_POPUP,^
rc.right - ROM_WIN_WIDTH, rc.bottom - ROM_WIN_HEIGHT,^ ROM_WIN_WIDTH,
ROM_WIN_HEIGHT,^ NULL,^ NULL,^ ghInst,^ NULL);^ if(ghWnd == NULL)^
return FALSE;^ if(lpszCmdLine[0] == NULL)^ return FALSE;^
if(HookSetup(lpszCmdLine) == FALSE)^ return FALSE;^ ButWnd =
CreateWindow("BUTTON", "Romulus",^ BS_PUSHBUTTON | WS_CHILD | WS_VISIBLE |
WS_TABSTOP,^ 0, -1, ROM_BUT_WIDTH, ROM_BUT_HEIGHT, ghWnd,^ ROM_BUTTON_ID,
ghInst, NULL);^ SetWindowPos(ghWnd, HWND_TOPMOST, 532, 0, 70, 20,
/*SWP_NOMOVE|*/ SWP_NOSIZE /*SWP_DRAWFRAME*/);^ ShowWindow(ghWnd,
SW_SHOWNOACTIVATE);^ UpdateWindow(ghWnd);^ while(GetMessage(&msg, NULL, 0, 0))
{^ TranslateMessage(&msg);^ DispatchMessage(&msg);^ }^ return msg.wParam;^ }^
BOOL HookSetup(LPSTR AppWindowName)^ {^ char Buffer[200] = { NULL };^ HWND
DeskWnd;^ FARPROC Proc;^ ghAppWnd = NULL;^ if(lstrcmp(AppWindowName,
WORD_WINDOW_NAME) == 0) {^ ghAppWnd = FuzzyFindWindow(WORD_MAIN_CLASS,
AppWindowName);^ gAppID = WORD_ID;^ lstrcpy(MainClass, WORD_MAIN_CLASS);^
lstrcpy(DeskClass, WORD_DESK_CLASS);^ lstrcpy(ChildClass, WORD_CHILD_CLASS);^
GetPrivateProfileString(LEAVEAPPSOPEN_SECTION, WP_INI_ENTRY, "",^ Buffer,
sizeof(Buffer), INI_NAME);^ SpecialClose = FALSE;^ } ^ else

```

- 31 -

```

if(lstrcmp(AppWindowName, WORDPERFECT_WINDOW_NAME) == 0) {^ ghAppWnd =
FuzzyFindWindow(WORDPERFECT_MAIN_CLASS, AppWindowName);^ gAppID =
WORDPERFECT_ID;^ lstrcpy(MainClass, WORDPERFECT_MAIN_CLASS);^
lstrcpy(DeskClass, WORDPERFECT_DESK_CLASS);^ lstrcpy(ChildClass,
WORDPERFECT_CHILD_CLASS);^ GetPrivateProfileString(LEAVEAPPSOPEN_SECTION,
WP_INI_ENTRY, "",^ Buffer, sizeof(Buffer), INI_NAME);^ SpecialClose =
FALSE;^ }^ else if(lstrcmp(AppWindowName, AMIPRO_WINDOW_NAME) == 0) {^
ghAppWnd = FuzzyFindWindow(AMIPRO_MAIN_CLASS, AppWindowName);^ gAppID =
AMIPRO_ID;^ lstrcpy(MainClass, AMIPRO_MAIN_CLASS);^ lstrcpy(DeskClass,
AMIPRO_DESK_CLASS);^ lstrcpy(ChildClass, AMIPRO_CHILD_CLASS);^
GetPrivateProfileString(LEAVEAPPSOPEN_SECTION, WP_INI_ENTRY, "",^ Buffer,
sizeof(Buffer), INI_NAME);^ SpecialClose = FALSE;^ }^ else
if(lstrcmp(AppWindowName, EXCEL_WINDOW_NAME) == 0) {^ ghAppWnd =
FuzzyFindWindow(EXCEL_MAIN_CLASS, AppWindowName);^ gAppID = EXCEL_ID;^
lstrcpy(MainClass, EXCEL_MAIN_CLASS);^ lstrcpy(DeskClass, EXCEL_DESK_CLASS);^
lstrcpy(ChildClass, EXCEL_CHILD_CLASS);^
GetPrivateProfileString(LEAVEAPPSOPEN_SECTION, SPREAD_INI_ENTRY, "",^
Buffer, sizeof(Buffer), INI_NAME);^ SpecialClose = FALSE;^ }^ else
if(lstrcmp(AppWindowName, LOTUS123_WINDOW_NAME) == 0) {^ ghAppWnd =
FuzzyFindWindow(LOTUS123_MAIN_CLASS, AppWindowName);^ gAppID = LOTUS123_ID;^
lstrcpy(MainClass, LOTUS123_MAIN_CLASS);^ lstrcpy(DeskClass,
LOTUS123_DESK_CLASS);^ lstrcpy(ChildClass, LOTUS123_CHILD_CLASS);^
GetPrivateProfileString(LEAVEAPPSOPEN_SECTION, SPREAD_INI_ENTRY, "",^
Buffer, sizeof(Buffer), INI_NAME);^ SpecialClose = FALSE;^ }^ else
if(lstrcmp(AppWindowName, QPRO_WINDOW_NAME) == 0) {^ ghAppWnd =
FuzzyFindWindow(QPRO_MAIN_CLASS, AppWindowName);^ gAppID = QPRO_ID;^
lstrcpy(MainClass, QPRO_MAIN_CLASS);^ lstrcpy(DeskClass, QPRO_DESK_CLASS);^
lstrcpy(ChildClass, QPRO_CHILD_CLASS);^
GetPrivateProfileString(LEAVEAPPSOPEN_SECTION, SPREAD_INI_ENTRY, "",^
Buffer, sizeof(Buffer), INI_NAME);^ SpecialClose = FALSE;^ }^ else
if(lstrcmp(AppWindowName, ABC_WINDOW_NAME) == 0) {^ ghAppWnd =
FuzzyFindWindow(ABC_MAIN_CLASS, AppWindowName);^ gAppID = ABC_ID;^
lstrcpy(MainClass, ABC_MAIN_CLASS);^ lstrcpy(DeskClass, ABC_DESK_CLASS);^
lstrcpy(ChildClass, ABC_CHILD_CLASS);^
GetPrivateProfileString(LEAVEAPPSOPEN_SECTION, CHART_INI_ENTRY, "",^ Buffer,
sizeof(Buffer), INI_NAME);^ SpecialClose = TRUE;^ }^ else
if(lstrcmp(AppWindowName, PSTYLER_WINDOW_NAME) == 0) {^ ghAppWnd =
FuzzyFindWindow(PSTYLER_MAIN_CLASS, AppWindowName);^ gAppID = PSTYLER_ID;^

```

- 32 -

```

lstrcpy(MainClass, PSTYLER_MAIN_CLASS);^ lstrcpy(DeskClass,
PSTYLER_DESK_CLASS);^ lstrcpy(ChildClass, PSTYLER_CHILD_CLASS);^
GetPrivateProfileString(LEAVEAPPSOPEN_SECTION, VIEWER_INI_ENTRY, "",^
Buffer, sizeof(Buffer), INI_NAME);^ SpecialClose = TRUE;^ }^ else
if(lstrcmp(AppWindowName, FTCOLOR_WINDOW_NAME) == 0) {^ ghAppWnd =
FuzzyFindWindow(FTCOLOR_MAIN_CLASS, AppWindowName);^ gAppID = FTCOLOR_ID;^
lstrcpy(MainClass, FTCOLOR_MAIN_CLASS);^ lstrcpy(DeskClass,
FTCOLOR_DESK_CLASS);^ lstrcpy(ChildClass, FTCOLOR_CHILD_CLASS);^
GetPrivateProfileString(LEAVEAPPSOPEN_SECTION, VIEWER_INI_ENTRY, "",^
Buffer, sizeof(Buffer), INI_NAME);^ SpecialClose = TRUE;^ /* start special
hook for FT Color */^ DeskWnd = NULL;^ Proc = MakeProcInstance(Desk_Proc,
ghInst);^ EnumChildWindows(ghAppWnd, Proc, (DWORD) ((HWND FAR *)&DeskWnd));^
FreeProcInstance(Proc);^ if((DeskWnd == NULL) || ((FTHOOKSET =
RomExtraSet(DeskWnd)) == FALSE))^ return FALSE; ^ }^ else
if(lstrcmp(AppWindowName, WORDSCAN_WINDOW_NAME) == 0) {^ ghAppWnd =
FuzzyFindWindow(WORDSCAN_MAIN_CLASS, AppWindowName);^ gAppID = WORDSCAN_ID;^
lstrcpy(MainClass, WORDSCAN_MAIN_CLASS);^ /*lstrcpy(DeskClass,
WORDSCAN_DESK_CLASS);*/^ lstrcpy(ChildClass, WORDSCAN_CHILD_CLASS);^
GetPrivateProfileString(LEAVEAPPSOPEN_SECTION, VIEWER_INI_ENTRY, "",^
Buffer, sizeof(Buffer), INI_NAME);^ SpecialClose = FALSE;^ }^ else
if(lstrcmp(AppWindowName, OPENIMAGE_WINDOW_NAME) == 0) {^ ghAppWnd =
FuzzyFindWindow(OPENIMAGE_MAIN_CLASS, AppWindowName);^ gAppID = OPENIMAGE_ID;^
lstrcpy(MainClass, OPENIMAGE_MAIN_CLASS);^ lstrcpy(DeskClass,
OPENIMAGE_DESK_CLASS);^ lstrcpy(ChildClass, OPENIMAGE_CHILD_CLASS);^
GetPrivateProfileString(LEAVEAPPSOPEN_SECTION, VIEWER_INI_ENTRY, "",^
Buffer, sizeof(Buffer), INI_NAME);^ SpecialClose = FALSE; ^ }^ else
if(lstrcmp(AppWindowName, ISYS_WINDOW_NAME) == 0) {^ ghAppWnd =
FuzzyFindWindow(ISYS_MAIN_CLASS, AppWindowName);^ gAppID = ISYS_ID;^
lstrcpy(MainClass, ISYS_MAIN_CLASS);^
GetPrivateProfileString(LEAVEAPPSOPEN_SECTION, TR_INI_ENTRY, "",^ Buffer,
sizeof(Buffer), INI_NAME);^ }^ else if(lstrcmp(AppWindowName,
ISYSUTILS_WINDOW_NAME) == 0) {^ ghAppWnd =
FuzzyFindWindow(ISYSUTILS_MAIN_CLASS, AppWindowName);^ gAppID = ISYSUTILS_ID;^
/* this app is always closed */^ }^ else if(lstrcmp(AppWindowName,
AVERY_WINDOW_NAME) == 0) {^ ghAppWnd = FuzzyFindWindow(NULL, AppWindowName);^
gAppID = AVERY_ID;^ /* this app is always closed */^ }^ else
if(lstrcmp(AppWindowName, ROMCALC_WINDOW_NAME) == 0) {^ ghAppWnd =
FuzzyFindWindow(ROMCALC_MAIN_CLASS, AppWindowName);^ gAppID = ROMCALC_ID;^ /*

```

```

this app is always closed */^ }^ else if(lstrcmp(AppWindowName,
LOTUSORG_WINDOW_NAME) == 0) {^ ghAppWnd = FuzzyFindWindow(LOTUSORG_MAIN_CLASS,
AppWindowName);^ gAppID = LOTUSORG_ID;^ /* this app is always closed */^ }^
else if(lstrcmp(AppWindowName, DOSBACKUP_WINDOW_NAME) == 0) {^ ghAppWnd =
FuzzyFindWindow(DOSBACKUP_MAIN_CLASS, AppWindowName);^ gAppID = DOSBACKUP_ID;^
/* this app is always closed */^ }^ gLeaveOpen = (lstrcmp(AnsiLower(Buffer),
TRUE_STRING) == 0)? TRUE : FALSE;^ if(ghAppWnd == NULL)^ return FALSE;^
HookSet = RomSet(ghAppWnd, ghWnd);^ if(HookSet == 0)^ return FALSE;^ return
TRUE;^ }^ long FAR PASCAL _export Main_Proc(HWND hWnd, unsigned Message,^
WORD wParam, LONG lParam)^ {^ switch (Message)^ {^ case WM_USER + 50:~ case
WM_DESTROY:~ if((HookSet) || (FTHookSet))^ RomFree();^ hWnd =
FindWindow("OMain", NULL);^ if(hWnd) ^ if(!IsZoomed(hWnd)) ShowWindow(hWnd,
SW_SHOWMAXIMIZED);^ else SetActiveWindow(hWnd);^ else {^ MessageBox(ghWnd,
"The Romulus MS Access Database has been closed.",^ AppName, MB_OK);^ }^
PostQuitMessage(0);^ break;^ case WM_USER + 51:~ CloseAndSwitch();^ break;^
case WM_COMMAND:~ switch(wParam) {^ case ROM_BUTTON_ID:~ CloseAndSwitch();^
break;^ }^ break;^ default:~ return DefWindowProc(hWnd, Message, wParam,
lParam);^ }^ return(0);^ }^ void CloseAndSwitch()^ {^
SetActiveWindow(ghAppWnd);^ if((gAppID == AVERY_ID) || (gAppID == ROMCALC_ID)
|| ^ (gAppID == ISYSUTILS_ID) || (gAppID == LOTUSORG_ID) || (gAppID ==
DOSBACKUP_ID)) ^ {^ if(IsWindow(ghAppWnd)) ^ PostMessage(ghAppWnd, WM_CLOSE,
0, 0);^ else ^ /* close romlink */^ DestroyWindow(ghWnd);^ }^ else if (gAppID
== ISYS_ID) {^ if(gLeaveOpen) {^ EnumWindows(ISYS_Proc, NULL);^ }^ else {^
PostMessage(ghAppWnd, WM_CLOSE, 0, 0);^ }^ DestroyWindow(ghWnd);^ } ^
if(CloseChildWindows() == FALSE) {^ if(gLeaveOpen == FALSE)^
PostMessage(ghAppWnd, WM_CLOSE, 0, 0);^ DestroyWindow(ghWnd);^ }^ } ^ }^ BOOL
CloseChildWindows()^ {^ HWND MainWnd, DeskWnd;^ BOOL Result = FALSE;^ FARPROC
Proc;^ MainWnd = FindWindow(MainClass, NULL);^ if(MainWnd) {^ DeskWnd = NULL;^
Proc = MakeProcInstance(Desk_Proc, ghInst);^ EnumChildWindows(MainWnd, Proc,
(DWORD) ((HWND FAR *)&DeskWnd));^ FreeProcInstance(Proc);^ if(DeskWnd) {^ Proc
= MakeProcInstance(Child_Proc, ghInst);^ EnumChildWindows(MainWnd, Proc,
(DWORD) (BOOL FAR *)&Result);^ FreeProcInstance(Proc);^ }^ }^ return(Result);^
}^ BOOL CALLBACK _export Desk_Proc(HWND hWndChild, HWND FAR *DeskWnd)^ {^ char
szClass[MAX_CLASS_LEN] = { NULL };^ GetClassName(hWndChild, szClass,
MAX_CLASS_LEN);^ if(lstrcmp(szClass, DeskClass) == 0) {^ *DeskWnd =
hWndChild;^ return(0);^ }^ return(1); ^ }^ BOOL CALLBACK _export
Child_Proc(HWND hWndChild, BOOL FAR *Result)^ {^ char szClass[MAX_CLASS_LEN] =
{ NULL };^ GetClassName(hWndChild, szClass, MAX_CLASS_LEN);^

```

- 34 -

```

if(lstrcmp(szClass, ChildClass) == 0) {^ RomSetChildWindow(hWndChild);^
if(SpecialClose)^ PostMessage(hWndChild, WM_SYSCOMMAND, SC_CLOSE, 0);^ else^
PostMessage(hWndChild, WM_CLOSE, 0, 0);^ *Result = TRUE;^ return(0);^ } ^
return(1); ^ }^ BOOL CALLBACK _export ISYS_Proc(HWND hWnd, LPARAM lParam)^ {^
char Buffer[MAX_STRING_LEN];^ GetClassName(hWnd, Buffer, MAX_STRING_LEN);^
if(lstrcmp(Buffer, ISYS_MAIN_CLASS) == 0) {^ GetWindowText(hWnd, Buffer,
lstrlen(ISYS_WINDOW_NAME) + 1);^ if( (lstrcmp(Buffer, ISYS_WINDOW_NAME) != 0)
&& (IsWindowVisible(hWnd)) )^ SendMessage(hWnd, WM_CLOSE, 0, 0);^ }^
return(1); ^ }^ HWND FuzzyFindWindow(LPSTR lpszClassName, LPSTR lpszWindow)^
{^ FUZZYPARAMS FuzzyParams;^ FARPROC Proc;^ FuzzyParams.lpszClassName =
lpszClassName;^ FuzzyParams.lpszWindow = lpszWindow;^ FuzzyParams.hWnd =
NULL;^ Proc = MakeProcInstance(Fuzzy_Proc, ghInst);^ EnumWindows(Proc, (DWORD)
((FUZZYPARAMS FAR *)&FuzzyParams));^ FreeProcInstance(Proc);^ return
FuzzyParams.hWnd;^ }^ BOOL CALLBACK _export Fuzzy_Proc(HWND hWnd, FUZZYPARAMS
FAR *lpFuzzyParams)^ {^ char Buffer[MAX_STRING_LEN];^ if(lpFuzzyParams->
lpszClassName != NULL) {^ GetClassName(hWnd, Buffer, MAX_STRING_LEN);^
if(lstrcmp(Buffer, lpFuzzyParams->lpszClassName) != 0)^ return (1);^ }^
if(lpFuzzyParams->lpszWindow != NULL) {^ GetWindowText(hWnd, Buffer,
sizeof(Buffer));^ if(_fstrchr(Buffer, lpFuzzyParams->lpszWindow) == NULL)^
return (1);^ }^ lpFuzzyParams->hWnd = hWnd; ^ return(0); ^ }^ ROMLINK.H^
#include <windows.h>^ #include <string.h>^ #include "romlrc.h"^ typedef struct
{^ LPSTR lpszClassName;^ LPSTR lpszWindow;^ HWND hWnd;^ } FUZZYPARAMS;^
typedef unsigned int near* NPUINT;^ typedef unsigned int far* LPUINT;^
typedef unsigned long near* NPULONG;^ typedef unsigned long far* LPULONG;^
typedef long NEAR* NPLONG;^ #define ROM_BUTTON_ID 17^ #define ROM_WIN_WIDTH
70^ #define ROM_WIN_HEIGHT 20^ #define ROM_BUT_WIDTH 70^ #define
ROM_BUT_HEIGHT 20^ #define MAX_CLASS_LEN 30^ #define MAX_STRING_LEN
256^ #define INI_NAME "romulus.ini" ^ #define LEAVEAPPSOPEN_SECTION
"LeaveApplicationsOpen"^ #define WP_INI_ENTRY "WordProcessor"^ #define
VIEWER_INI_ENTRY "Viewer"^ #define TR_INI_ENTRY "TextRetrieval"^ #define
SPREAD_INI_ENTRY "SpreadSheet"^ #define CHART_INI_ENTRY "Chart"^ #define
TRUE_STRING "true"^ /*#define WORD_NAME "Winword"*/^ #define
WORD_WINDOW_NAME "Microsoft Word"^ #define WORD_MAIN_CLASS "OpusApp"^
#define WORD_DESK_CLASS "OpusDesk"^ #define WORD_CHILD_CLASS "OpusMwd"^
#define WORD_ID 1^ /*#define ABC_NAME "abc 2"*/^ #define
ABC_WINDOW_NAME "ABC FlowCharter"^ #define ABC_MAIN_CLASS "Ren &
Stimpy"^ #define ABC_DESK_CLASS "MDIClient"^ #define ABC_CHILD_CLASS
"Ren Hoek"^ #define ABC_ID 2 ^ /*#define ABC_NAME "abc 3"*/^ #define

```

- 35 -

```

ABC_WINDOW_NAME      "Micrografx ABC FlowCharter"^^ #define ABC_MAIN_CLASS
"ABCFrame"^^ #define ABC_DESK_CLASS      "MDIClient"^^ #define ABC_CHILD_CLASS
"ABCChart"^^ #define ABC_ID              2^ /*#define EXCEL_4_NAME      "excel"*/^
#define EXCEL_WINDOW_NAME "Microsoft Excel"^^ #define EXCEL_MAIN_CLASS
"XLMAIN"^^ #define EXCEL_DESK_CLASS      "XLDESK"^^ #define EXCEL_CHILD_CLASS
"EXCEL5"^^ #define EXCEL_ID              3^ /*#define EXCEL_5_NAME      "excel"*/^ #define
EXCEL_WINDOW_NAME "Microsoft Excel"^^ #define EXCEL_MAIN_CLASS "XLMAIN"^^
#define EXCEL_DESK_CLASS "XLDESK"^^ #define EXCEL_CHILD_CLASS "EXCEL9"^^
#define EXCEL_ID        3^ /*#define PSTYLER_NAME      "pstyler"*/^ #define
PSTYLER_WINDOW_NAME "PhotoStyler Special Edition"^^ #define PSTYLER_MAIN_CLASS
"Frame"^^ #define PSTYLER_DESK_CLASS      "MDIClient"^^ #define PSTYLER_CHILD_CLASS
"MDIChild"^^ #define PSTYLER_ID           4^ /*#define FTCOLOR_NAME 1.0
"ftcolor"*/^ #define FTCOLOR_WINDOW_NAME "FotoTouch Color"^^ #define
FTCOLOR_MAIN_CLASS "appwindow"^^ #define FTCOLOR_DESK_CLASS "MDIClient"^^
#define FTCOLOR_CHILD_CLASS "mdichildwindow"^^ #define FTCOLOR_ID      5 ^
/*#define FTCOLOR_NAME 1.0a "ftcolor"*/^ #define FTCOLOR_WINDOW_NAME
"FotoTouch Color"^^ #define FTCOLOR_MAIN_CLASS "FTColor"^^ #define
FTCOLOR_DESK_CLASS "MDIClient"^^ #define FTCOLOR_CHILD_CLASS
"mdichildwindow"^^ #define FTCOLOR_ID      5 ^ /*#define ISYS_NAME      "iq"*/^
#define ISYS_WINDOW_NAME "ISYS Query"^^ #define ISYS_MAIN_CLASS
"ThunderForm"^^ #define ISYS_ID            6^ /*#define ISYSUTILS_NAME "idbw"*/^
#define ISYSUTILS_WINDOW_NAME "ISYS Utilities"^^ #define ISYSUTILS_MAIN_CLASS
"ThunderForm"^^ #define ISYSUTILS_ID       7^ /*#define AVERY_NAME      "lpwin"*/^
#define AVERY_WINDOW_NAME "LabelPro"^^ #define AVERY_ID          8^ /*#define
ROMCALC_NAME "romcalc"*/^ #define ROMCALC_MAIN_CLASS "RomCalc10"^^ #define
ROMCALC_WINDOW_NAME "Romulus Resource Calculator"^^ #define ROMCALC_ID    9^
/*#define ORGANISER_NAME "organize"*/^ #define LOTUSORG_MAIN_CLASS
"TZ_PORG"^^ #define LOTUSORG_WINDOW_NAME "Lotus Organizer"^^ #define
LOTUSORG_ID        10^ /*#define WORDSCAN_NAME      "wordscan 2"*/^ #define
WORDSCAN_WINDOW_NAME "WordScan Plus - JOB01"^^ #define WORDSCAN_MAIN_CLASS
"Calera.WordScan"^^ #define WORDSCAN_CHILD_CLASS "Calera.WordScan.Viewer"^^
#define WORDSCAN_ID    11 ^ /*#define WORDPERFECT_NAME "wordperfect
6.0"*/^ #define WORDPERFECT_WINDOW_NAME "WordPerfect"^^ #define
WORDPERFECT_MAIN_CLASS "WordPerfect"^^ #define WORDPERFECT_DESK_CLASS
"MDIClient"^^ #define WORDPERFECT_CHILD_CLASS "WPDocFrame"^^ #define
WORDPERFECT_ID      12^ /*#define AMIPRO_NAME      "Ami Pro 3.0"*/^ #define
AMIPRO_WINDOW_NAME "Ami Pro"^^ #define AMIPRO_MAIN_CLASS "AmiProWndA"^^
#define AMIPRO_DESK_CLASS "MDIClient"^^ #define AMIPRO_CHILD_CLASS

```

- 36 -

```

"MDICCHILD" ^ #define AMIPRO_ID      13 ^ /* #define lotus123_NAME      "lotus 123
4.01" */ ^ #define LOTUS123_WINDOW_NAME "Lotus 1-2-3 Release 4" ^ #define
LOTUS123_MAIN_CLASS      "123WParent" ^ #define LOTUS123_DESK_CLASS
"123WMDIClient" ^ #define LOTUS123_CHILD_CLASS "123WSheet" ^ #define
LOTUS123_ID      14 ^ /* #define QPRO_NAME      "Quattro pro 5.0" */ ^ #define
QPRO_WINDOW_NAME "Quattro Pro for Windows" ^ #define QPRO_MAIN_CLASS
"QPWBaseWdw" ^ #define QPRO_DESK_CLASS      "MDIClient" ^ #define QPRO_CHILD_CLASS
"QPWChildWdw" ^ #define QPRO_ID      15 ^ /* #define DOSBACKUP_NAME      "Backup
v6" */ ^ #define DOSBACKUP_WINDOW_NAME "Microsoft Backup -" ^ #define
DOSBACKUP_MAIN_CLASS "MWBACKUP" ^ #define DOSBACKUP_ID      16 ^ /* #define
OPENIMAGE_NAME      "OPEN/image" */ ^ #define OPENIMAGE_WINDOW_NAME "OPEN/image
Cabinet" ^ #define OPENIMAGE_MAIN_CLASS "WIISDisplay" ^ #define
OPENIMAGE_DESK_CLASS "ImageWindow" ^ #define OPENIMAGE_CHILD_CLASS
"ImageWindow" ^ #define OPENIMAGE_ID      17 ^ /* FUNCTION DECLARATIONS */ ^ int
PASCAL WinMain(HANDLE, HANDLE, LPSTR, int); ^ BOOL HookSetup(LPSTR); ^ long
__export FAR PASCAL Main_Proc(HWND, unsigned, WORD, LONG); ^ void
CloseAndSwitch(void); ^ BOOL CloseAppFile(void); ^ BOOL
CloseChildWindows(void); ^ BOOL __export CALLBACK Desk_Proc(HWND, HWND FAR *); ^
BOOL __export CALLBACK Child_Proc(HWND, BOOL FAR *); ^ BOOL __export CALLBACK
ISYS_Proc(HWND, LPARAM); ^ HWND FuzzyFindWindow(LPSTR, LPSTR); ^ BOOL __export
CALLBACK Fuzzy_Proc(HWND, FUZZYPARAMS FAR *); ^ long __export FAR PASCAL
SubClass_Proc(HWND, unsigned, WORD, LONG); ^ extern HANDLE ghInst; ^ extern HWND
ghWnd, ButWnd; ^ extern char AppName[200]; ^ BOOL __export CALLBACK RomSet(HWND,
HWND); ^ BOOL __export CALLBACK RomExtraSet(HWND); ^ void __export CALLBACK
RomFree(void); ^ void __export CALLBACK RomSetChildWindow(HWND); ^ EXTERNAL.H
int PASCAL WinMain(HANDLE, HANDLE, LPSTR, int); ^ long FAR PASCAL __export
Main_Proc(HWND, unsigned, WORD, LONG); ^ void CloseAndSwitch(void); ^ BOOL
CloseAppFile(void); ^ BOOL CloseChildWindows(void); ^ BOOL CALLBACK __export
Desk_Proc(HWND, HWND FAR *); ^ BOOL CALLBACK __export Child_Proc(HWND, BOOL FAR
*); ^ BOOL CALLBACK __export ISYS_Proc(HWND, LPARAM); ^ HWND
FuzzyFindWindow(LPSTR, LPSTR); ^ BOOL CALLBACK __export Fuzzy_Proc(HWND,
FUZZYPARAMS FAR *); ^ extern HANDLE ghInst; ^ extern HWND ghWnd, ButWnd; ^ extern
char AppName[200]; ^ ROMLINK.DEF ^ NAME RomLink ^ DESCRIPTION 'Romulus RomLink.
Copyright (c) 1993 - 1995 Ni-Tech Pty Limited. Patent Pending.' ^ EXETYPE
WINDOWS ^ STUB 'WINSTUB.EXE' ^ CODE MOVEABLE PRELOAD ^ DATA MOVEABLE MULTIPLE
PRELOAD ^ HEAPSIZE 8192 ^ INITIATING FILES ^ '* '* Romulus - Global Module ^ '*
Copyright 1993 Romulus Software ^ '* Option Compare Database 'Use database
order for string comparisons ^ Option Explicit ^ ' * Global document and image

```

- 37 -

```

paths^ Global Rom_DocumentPath As String^ Global Rom_ImagePath As String^
Global ROM_ISYSTEXTPATH As String^ Global Rom_SpreadSheetPath As String^
Global Rom_ChartPath As String^ ' * this name will be used in all dialog boxes
^^ Global Const ROMULUS_APP_NAME = "Romulus"^^ ' * some ini names ^^ Global
Const ROM_INI_NAME = "romulus.ini"^^ Global Const ROM_APP_INI_SECTION =
"Applications"^^ Global Const ROMLINK_INI_ENTRY = "RomulusLink"^^ Global Const
ROM_WP_INI_ENTRY = "WordProcessor"^^ Global Const ROM_WP2_INI_ENTRY =
"WordProcessor2"^^ Global Const ROM_WP3_INI_ENTRY = "WordProcessor3"^^ Global
Const ROM_VIEWER_INI_ENTRY = "Viewer"^^ Global Const ROM_VIEWER2_INI_ENTRY =
"Viewer2"^^ Global Const ROM_VIEWER3_INI_ENTRY = "Viewer3"^^ Global Const
ROM_TR_INI_ENTRY = "TextRetrieval"^^ Global Const ROM_TRUTILS_INI_ENTRY =
"TextRetrievalUtilities"^^ Global Const ROM_SPREAD_INI_ENTRY = "SpreadSheet"^^
Global Const ROM_SPREAD2_INI_ENTRY = "SpreadSheet2"^^ Global Const
ROM_SPREAD3_INI_ENTRY = "SpreadSheet3"^^ Global Const ROM_CHART_INI_ENTRY =
"Chart"^^ Global Const ROM_LABEL_INI_ENTRY = "Labels"^^ Global Const
ROMCALC_INI_ENTRY = "RomulusCalc"^^ Global Const ROM_VOICE_INI_ENTRY =
"VoiceControl"^^ Global Const ROM_READBACK_INI_ENTRY = "AudioReadBack"^^ Global
Const ROM_JOIN_INI_ENTRY = "JoinTemplate"^^ Global Const ROM_CONVERT_INI_ENTRY
= "ConvertTemplate"^^ Global Const ROM_TEXTSAVE_INI_ENTRY = "TextSaveTemplate"^^
Global Const ROM_ORG_INI_ENTRY = "Organizer"^^ Global Const
ROM_BACKUP_INI_ENTRY = "Backup"^^ Global Const ROM_MAX_STRINGLEN = 256^ ' *
application main window classes ^^ Global Const ROM_WORD_WINDOWCLASS =
"OpusApp"^^ Global Const ROM_PSTYLER_WINDOWCLASS = "Frame"^^ Global Const
ROM_WORDPERFECT_WINDOWCLASS = "WordPerfect"^^ Global Const
ROM_AMIPRO_WINDOWCLASS = "AmiProWnda"^^ Global Const ROM_FTCCOLOR_WINDOWCLASS =
"FTColor"^^ Global Const ROM_EXCEL_WINDOWCLASS = "XLMAIN"^^ Global Const
ROM_LOTUS123_WINDOWCLASS = "123WParent"^^ Global Const ROM_QPRO_WINDOWCLASS =
"QPWBaseWdw"^^ Global Const ROM_ABC_WINDOWCLASS = "ABCFrame"^^ Global Const
ROM_ISYS_WINDOWCLASS = "ThunderForm"^^ Global Const ROM_AVERY_WINDOWCLASS = ""^^
Global Const ROMCALC_WINDOW_CLASS = "RomCalc10"^^ Global Const
ROM_ISYSUTILS_WINDOWCLASS = "ThunderForm"^^ Global Const ROM_ORG_WINDOWCLASS =
"TZ_PORG"^^ Global Const ROM_WORDSCAN_WINDOWCLASS = "Calera.WordScan"^^ Global
Const ROM_OPENIMAGE_WINDOWCLASS = "WIISDisplay"^^ Global Const
ROM_BACKUP_WINDOWCLASS = "MWBACUP"^^ Global Const ROM_WORD_WINDOWNAME =
"Microsoft Word"^^ Global Const ROM_PSTYLER_WINDOWNAME = "PhotoStyler Special
Edition"^^ Global Const ROM_WORDPERFECT_WINDOWNAME = "WordPerfect"^^ Global
Const ROM_AMIPRO_WINDOWNAME = "Ami Pro"^^ Global Const ROM_FTCCOLOR_WINDOWNAME =
"FotoTouch Color"^^ Global Const ROM_EXCEL_WINDOWNAME = "Microsoft Excel"^^

```

- 38 -

```

Global Const ROM_LOTUS123_WINDOWNAME = "Lotus 1-2-3 Release 4" Global Const
ROM_QPRO_WINDOWNAME = "Quattro Pro for Windows" Global Const
ROM_ABC_WINDOWNAME = "Micrografx ABC FlowCharter" Global Const
ROM_ISYS_WINDOWNAME = "ISYS Query" Global Const ROM_AVERY_WINDOWNAME =
"LabelPro" Global Const ROMCALC_WINDOW_NAME = "Romulus Resource Calculator"
Global Const ROM_ISYSUTILS_WINDOWNAME = "ISYS Utilities" Global Const
ROM_ORG_WINDOWNAME = "Lotus Organizer" Global Const ROM_WORDSCAN_WINDOWNAME =
"WordScan Plus - JOB01" Global Const ROM_OPENIMAGE_WINDOWNAME = "OPEN/image
Cabinet" Global Const ROM_BACKUP_WINDOWNAME = "Microsoft Backup -" Global
Const ROM_ISYS_AINDEX = "ISYS.IXA" ' * Windows API ShowWindow() defines *
Global Const ROM_SW_SHOWMAXIMIZED = 3 ' * message box defines * msgbox
Global Const ROM_MB_OK = 0 Global Const ROM_MB_YESNO = 4 Global Const
ROM_MB_ICONEXCLAMATION = 48 Global Const ROM_MB_ICONINFORMATION = 64 Global
Const ROM_MB_ICONQUESTION = 32 Global Const MB_DEFBUTTON2 = 256, IDYES = 6,
IDNO = 7 ' * ini entries to hold the file name to store numbers in Global
Const ROM_NUMFILE_INI_SECTION = "Numbers" Global Const ROM_NUMFILE_INI_ENTRY
= "TextFile" Global Const ROM_BARFILE_INI_ENTRY = "BarFile" Global Const
ROM_PLAINFILE_INI_ENTRY = "PlainFile" ' * ini entry in avery labels of the
last loaded label form Global Const ROM_AVERY_LAYOUT_SECTION = "Layout
Editor" Global Const ROM_AVERY_LASTFILE_ENTRY = "LastSavedLayout" Global
Const ROM_AVERY_INI_NAME = "lpwin.ini" ' * Windows API WM_CLOSE Global Const
ROM_WM_CLOSE = 16 Global Const ROM_WM_SYSCOMMAND = 274 Global Const
ROM_SC_CLOSE = -4000 ' * DLL function declarations Declare Function
Rom_FindWindow Lib "user" Alias "FindWindow" (ByVal ClassName As String, ByVal
Title As Long) As Integer Declare Function Rom_ShowWindow Lib "user" Alias
"ShowWindow" (ByVal WindowHand As Integer, ByVal ShowType As Integer) As
Integer Declare Function Rom_SetActiveWindow Lib "user" Alias
"SetActiveWindow" (ByVal WindowHand As Integer) As Integer Declare Function
Rom_GetActiveWindow Lib "user" Alias "GetActiveWindow" () As Integer Declare
Function Rom_GetModuleHandle Lib "kernel" Alias "GetModuleHandle" (ByVal
lpAppName As String) As Integer Declare Function Rom_IsZoomed Lib "user"
Alias "IsZoomed" (ByVal WindowHand As Integer) As Integer Declare Sub
Rom_Yield Lib "kernel" Alias "Yield" () Declare Function
Rom_GetPrivateProfileString Lib "Kernel" Alias "GetPrivateProfileString"
(ByVal lpAppName As String, ByVal lpKeyName As String, ByVal lpDefault As
String, ByVal lpReturnedString As String, ByVal nSize As Integer, ByVal
lpFileName As String) As Integer Declare Function Rom_PostMessage Lib "user"
Alias "PostMessage" (ByVal hWnd As Integer, ByVal msg As Integer, ByVal wParam

```

- 39 -

```

As Integer, lParam As Long) As Integer^ 'Declare Function Rom_SendMessage Lib
"user" Alias "SendMessage" (ByVal hWnd As Integer, ByVal msg As Integer, ByVal
wParam As Integer, lParam As Long) As Integer^ Declare Function
Rom_WritePrivateProfileString Lib "Kernel" Alias "WritePrivateProfileString"
(ByVal lpAppName As String, ByVal lpKeyName As String, ByVal SentString As
String, ByVal lpFileName As String) As Integer^ Declare Function
Rom_FuzzyFindWindow Lib "romulus.dll" Alias "FuzzyFindWindow" (ByVal
lpzClassName As String, ByVal lpzWindow As String) As Integer^ Function
Rom_ReturnIniString (Section As String, Entry As String) As String^ On Error
GoTo NoString_Err^ Dim lpIniRetString As String^ Dim ret As Integer^ 'Create a
buffer for the returned ini string^ lpIniRetString =
Space$(ROM_MAX_STRINGLEN)^ ret = Rom_GetPrivateProfileString(Section, Entry,
"", lpIniRetString, ROM_MAX_STRINGLEN, ROM_INI_NAME)^ 'Truncate the fixed-
length string to the left of the Null terminator^ lpIniRetString =
Left$(lpIniRetString, (InStr(lpIniRetString, Chr$(0)) - 1))^
Rom_ReturnIniString = lpIniRetString^ RIS_Exit:^ Exit Function^ NoString_Err:^
Rom_ReturnIniString = ""^ Resume RIS_Exit^ End Function^ '** * Romulus -
OpenFileRoutines Module^ '* Copyright 1993 Romulus Software^ '** Option
Compare Database 'Use database order for string comparisons^ Option Explicit^
Function Rom_OpenABC ()^ On Error GoTo NoOpenABC_Err^ Dim ret As Integer^ ret
= Rom_OpenApp(ROM_CHART_INI_ENTRY, ROM_ABC_WINDOWCLASS, ROM_ABC_WINDOWNAME)^
ABC_Exit:^ Exit Function^ NoOpenABC_Err:^ MsgBox "Cannot open ABC Flowcharter.
Error: " + Error$, ROM_MB_OK + ROM_MB_ICONEXCLAMATION, ROMULUS_APP_NAME^
Resume ABC_Exit^ End Function^ Function Rom_OpenABCFile (Filename As String)^
On Error GoTo ABCFile_Err^ Dim ret As Integer^ ' * see if the filename is
valid^ Filename = Rom_ChartPath & Filename^ If Dir$(Filename) = "" Then^ ret =
MsgBox("There is no associated flowchart. Would you like to create a
flowchart?", ROM_MB_YESNO + MB_DEFBUTTON2 + ROM_MB_ICONQUESTION,
ROMULUS_APP_NAME)^ If ret = 7 Then GoTo ABCFile_Exit^ DoCmd RunMacro
"Indicator"^ ret = Rom_OpenApp(ROM_CHART_INI_ENTRY, ROM_ABC_WINDOWCLASS,
ROM_ABC_WINDOWNAME)^ SendKeys "%fs" & Filename & "{ENTER}"^ Exit Function^ End
If^ ' * open ABC or make it the active window^ ret =
Rom_OpenApp(ROM_CHART_INI_ENTRY, ROM_ABC_WINDOWCLASS, ROM_ABC_WINDOWNAME)^ ' *
open the filename^ If ret = True Then^ SendKeys "%fo" & Filename & "{ENTER}"^
End If^ ABCFile_Exit:^ Exit Function^ ABCFile_Err:^ MsgBox "Cannot open ABC
flowchart. Error: " + Error$, ROM_MB_OK + ROM_MB_ICONEXCLAMATION,
ROMULUS_APP_NAME^ Resume ABCFile_Exit^ End Function^ Function Rom_OpenAmiPro
()^ On Error GoTo NoOpenAmiPro_Err^ Dim ret As Integer^ ret =

```

```

Rom_OpenApp(ROM_WP3_INI_ENTRY, ROM_AMIPRO_WINDOWCLASS, ROM_AMIPRO_WINDOWNAME)^
AmiPro_Exit:^ Exit Function^ NoOpenAmiPro_Err:^ MsgBox "Cannot open Ami Pro.
Error: " + Error$, ROM_MB_OK + ROM_MB_ICONEXCLAMATION, ROMULUS_APP_NAME^
Resume AmiPro_Exit^ End Function^ Function Rom_OpenAmiProFile (Filename As
String)^ On Error GoTo APFile_Err^ Dim ret As Integer^ ' * see if the filename
is valid^ Filename = Rom_DocumentPath & Filename^ If Dir$(Filename) = "" Then^
ret = MsgBox("There is no associated document. Would you like to create a
document?", ROM_MB_YESNO + MB_DEFBUTTON2 + ROM_MB_ICONINFORMATION,
ROMULUS_APP_NAME)^ If ret = 7 Then GoTo APFile_Exit^ DoCmd RunMacro
"Indicator"^ ret = Rom_OpenApp(ROM_WP3_INI_ENTRY, ROM_AMIPRO_WINDOWCLASS,
ROM_AMIPRO_WINDOWNAME)^ SendKeys "%fn" & "{ENTER}"^ SendKeys "%fa" & Filename
& "{ENTER}"^ Exit Function^ End If^ ' * open Word or make it the active
window^ ret = Rom_OpenApp(ROM_WP3_INI_ENTRY, ROM_AMIPRO_WINDOWCLASS,
ROM_AMIPRO_WINDOWNAME)^ ' * open the filename^ If ret = True Then^ SendKeys
"%fo" & Filename & "{ENTER}"^ End If^ APFile_Exit:^ Exit Function^
APFile_Err:^ MsgBox "Cannot open Ami Pro document. Error: " + Error$,
ROM_MB_OK + ROM_MB_ICONEXCLAMATION, ROMULUS_APP_NAME^ Resume APFile_Exit^ End
Function^ Function Rom_OpenApp (AppIniEntry As String, AppClass As String,
AppWindow As String) As Integer^ On Error GoTo Open_Err^ ' Set up error
handler.^ Rom_OpenApp = True^ Dim WindowHandle As Integer^ Dim ret As Integer^
Dim lpIniRetString As String^ Dim AppName As String^ Dim RomName As String^
Dim AppModuleName As String^ Dim ParamPos As Integer^ ' * find the filename of
the app from the ini_file^ AppName = Rom_ReturnIniString(ROM_APP_INI_SECTION,
AppIniEntry)^ If (AppName = "") Then^ MsgBox "There is no filename listed for
" & AppWindow & "in the ROMULUS.INI file.", ROM_MB_OK +
ROM_MB_ICONEXCLAMATION, ROMULUS_APP_NAME^ Rom_OpenApp = False^ GoTo Open_Exit^
End If^ ' * load the app if it is not already loaded^ ' * to find the module
ignore any command line parameters in the name^ ParamPos = InStr(AppName, "
")^ If (ParamPos) Then^ AppModuleName = Left$(AppName, ParamPos - 1)^ Else^
AppModuleName = AppName^ End If^ If (Rom_GetModuleHandle(AppModuleName) = 0)
Then 'App is not loaded^ ret = Shell(AppName, 1)^ 'start App^ If ret = 0 Then
GoTo Open_Err^ ' * wait until the main window of the app has been displayed^
???? if it does not start up correctly we get stuck here forever^ While
(Rom_FuzzyFindWindow(AppClass, AppWindow) = 0)^ Rom_Yield^ Wend^ End If^ ' *
make sure we can find the window handle before we open Rom link^ WindowHandle
= Rom_FuzzyFindWindow(AppClass, AppWindow)^ If WindowHandle = 0 Then^ MsgBox
"Cannot Access " & AppWindow, ROM_MB_OK + ROM_MB_ICONEXCLAMATION,
ROMULUS_APP_NAME^ Rom_OpenApp = False^ GoTo Open_Exit^ End If^ ' * Start the

```

- 41 -

```

Romulus floating top button window^ ' * If it doesn't start we still let them
go ahead^ RomName = Rom_ReturnIniString(ROM_APP_INI_SECTION,
ROMLINK_INI_ENTRY)^ RomName = RomName & " " & AppWindow^ If (RomName = "")
Then^ MsgBox "There is no filename listed for the Romulus link program in the
ROMULUS.INI file.", ROM_MB_OK + ROM_MB_ICONEXCLAMATION, ROMULUS_APP_NAME^
Else^ ret = Shell(RomName, 8) 'start Romulus floating top window.^ End If^ ' *
maximize the window and make it the active window^ If
(Rom_IsZoomed(WindowHandle) = 0) Then^ ret = Rom_ShowWindow(WindowHandle,
ROM_SW_SHOWMAXIMIZED)^ Else^ ret = Rom_SetActiveWindow(WindowHandle)^ End If^
Open_Exit:^ Exit Function^ Open_Err:^ MsgBox "Cannot open " & AppWindow & ".
Error: " + Error$, ROM_MB_OK + ROM_MB_ICONEXCLAMATION, ROMULUS_APP_NAME^
Rom_OpenApp = False^ Resume Open_Exit^ End Function^ Function Rom_OpenExcel
(^ On Error GoTo NoOpenExcel_Err^ Dim ret As Integer^ ret =
Rom_OpenApp(ROM_SPREAD_INI_ENTRY, ROM_EXCEL_WINDOWCLASS,
ROM_EXCEL_WINDOWNAME)^ Excel_Exit:^ Exit Function^ NoOpenExcel_Err:^ MsgBox
"Cannot open Excel. Error: " + Error$, ROM_MB_OK + ROM_MB_ICONEXCLAMATION,
ROMULUS_APP_NAME^ Resume Excel_Exit^ End Function^ Function Rom_OpenExcelFile
(Filename As String)^ On Error GoTo ExcelFile_Err^ Dim ret As Integer^ ' * see
if the filename is valid^ Filename = Rom_SpreadSheetPath & Filename^ If
Dir$(Filename) = "" Then^ ret = MsgBox("There is no associated spreadsheet.
Would you like to create a spreadsheet?", ROM_MB_YESNO + MB_DEFBUTTON2 +
ROM_MB_ICONQUESTION, ROMULUS_APP_NAME)^ If ret = 7 Then GoTo ExcelFile_Exit^
DoCmd RunMacro "Indicator"^ ret = Rom_OpenApp(ROM_SPREAD_INI_ENTRY,
ROM_EXCEL_WINDOWCLASS, ROM_EXCEL_WINDOWNAME)^ SendKeys "%fa" & Filename &
"{ENTER}"^ Exit Function^ End If^ ' * open Excel or make it the active window^
ret = Rom_OpenApp(ROM_SPREAD_INI_ENTRY, ROM_EXCEL_WINDOWCLASS,
ROM_EXCEL_WINDOWNAME)^ ' * open the filename^ If ret = True Then^ SendKeys
"%fo" & Filename & "{ENTER}"^ End If^ ExcelFile_Exit:^ Exit Function^
ExcelFile_Err:^ MsgBox "Cannot open Excel spreadsheet. Error: " + Error$,
ROM_MB_OK + ROM_MB_ICONEXCLAMATION, ROMULUS_APP_NAME^ Resume ExcelFile_Exit^
End Function^ Function Rom_OpenFTColor (^ On Error GoTo NoOpenFTC_Err^ Dim
ret As Integer^ ret = Rom_OpenApp(ROM_VIEWER3_INI_ENTRY,
ROM_FTCOLOR_WINDOWCLASS, ROM_FTCOLOR_WINDOWNAME)^ FTC_Exit:^ Exit Function^
NoOpenFTC_Err:^ MsgBox "Cannot open FotoTouch Color. Error: " + Error$,
ROM_MB_OK + ROM_MB_ICONEXCLAMATION, ROMULUS_APP_NAME^ Resume FTC_Exit^ End
Function^ Function Rom_OpenFTColorFile (Filename As String)^ On Error GoTo
FTCFile_Err^ Dim ret As Integer^ ' * see if the filename is valid^ Filename =
Rom_ImagePath & Filename^ If Dir$(Filename) = "" Then^ ret = MsgBox("There is

```

- 42 -

```

no associated image. Would you like to create an image?", ROM_MB_YESNO +
MB_DEFBUTTON2 + ROM_MB_ICONQUESTION, ROMULUS_APP_NAME)^ If ret = 7 Then GoTo
FTCFFile_Exit^ DoCmd RunMacro "Indicator"^^ ret =
Rom_OpenApp(ROM_VIEWER3_INI_ENTRY, ROM_FTCOLOR_WINDOWCLASS,
ROM_FTCOLOR_WINDOWNAME)^ SendKeys "%fnl" & "{ENTER}"^^ SendKeys "%fa" &
Filename & "{ENTER}"^^ SendKeys "%fq"^^ Exit Function^ End If^ ' * open FT Color
or make it the active window^ ret = Rom_OpenApp(ROM_VIEWER3_INI_ENTRY,
ROM_FTCOLOR_WINDOWCLASS, ROM_FTCOLOR_WINDOWNAME)^ ' * open the filename^ If
ret = True Then^ SendKeys "%fo" & Filename & "{ENTER}"^^ End If^ FTCFile_Exit:^
Exit Function^ FTCFile_Err:^ MsgBox "Cannot open FotoTouch image. Error: " +
Error$, ROM_MB_OK + ROM_MB_ICONEXCLAMATION, ROMULUS_APP_NAME^ Resume
FTCFFile_Exit^ End Function^ Function Rom_OpenISYS (AppIniEntry As String,
AppClass As String, AppWindow As String) As Integer^ On Error GoTo ISYS_Err^
Set up error handler.^ Rom_OpenISYS = True^ Dim WindowHandle As Integer^ Dim
ret As Integer^ Dim lpIniRetString As String^ Dim AppName As String^ Dim
RomName As String^ Dim AppModuleName As String^ Dim ParamPos As Integer^ ' *
find the filename of the app from the ini_file^ AppName =
Rom_ReturnIniString(ROM_APP_INI_SECTION, AppIniEntry)^ If (AppName = "") Then^
MsgBox "There is no filename listed for " & AppWindow & "in the ROMULUS.INI
file.", ROM_MB_OK + ROM_MB_ICONEXCLAMATION, ROMULUS_APP_NAME^ Rom_OpenISYS =
False^ GoTo ISYS_Exit^ End If^ ' * load the app if it is not already loaded^
' * to find the module ignore any command line parameters in the name^ ParamPos
= InStr(AppName, " ")^ If (ParamPos) Then^ AppModuleName = Left$(AppName,
ParamPos - 1)^ Else^ AppModuleName = AppName^ End If^ If
(Rom_GetModuleHandle(AppModuleName) = 0) Then 'App is not loaded^ ret =
Shell(AppName, 1)^ 'start App^ If ret = 0 Then GoTo ISYS_Err^ ' * wait until
the main window of the app has been displayed^ ' ??? if it does not start up
correctly we get stuck here forever^ While (Rom_FuzzyFindWindow(AppClass,
AppWindow) = 0)^ Rom_Yield^ Wend^ SendKeys "%fc%ff" & "{DEL}" & "{ENTER}"^^
SendKeys "%fd" & "{TAB}" & "{TAB}"^^ SendKeys "+{END}"^^ SendKeys "{DEL}" &
Rom_ISYSXPath & "{ENTER}"^^ End If^ ' * make sure we can find the window
handle before we open Rom link^ WindowHandle = Rom_FuzzyFindWindow(AppClass,
AppWindow)^ If WindowHandle = 0 Then^ MsgBox "Cannot Access " & AppWindow,
ROM_MB_OK + ROM_MB_ICONEXCLAMATION, ROMULUS_APP_NAME^ Rom_OpenISYS = False^
GoTo ISYS_Exit^ End If^ ' * Start the Romulus floating top button window^ ' *
If it doesn't start we still let them go ahead^ RomName =
Rom_ReturnIniString(ROM_APP_INI_SECTION, ROMLINK_INI_ENTRY)^ RomName = RomName
& " " & AppWindow^ If (RomName = "") Then^ MsgBox "There is no filename listed

```

- 43 -

```

for the Romulus link program in the ROMULUS.INI file.", ROM_MB_OK +
ROM_MB_ICONEXCLAMATION, ROMULUS_APP_NAME^ Else^ ret = Shell(RomName, 8) 'start
Romulus floating top window.^ End If^ ' * maximize the window and make it the
active window^ If (Rom_IsZoomed(WindowHandle) = 0) Then^ ret =
Rom_ShowWindow(WindowHandle, ROM_SW_SHOWMAXIMIZED)^ Else^ ret =
Rom_SetActiveWindow(WindowHandle)^ End If^ ISYS_Exit:^ Exit Function^
ISYS_Err:^ MsgBox "Cannot open " & AppWindow & ". Error: " + Error$, ROM_MB_OK
+ ROM_MB_ICONEXCLAMATION, ROMULUS_APP_NAME^ Rom_OpenISYS = False^ Resume
ISYS_Exit^ End Function^ Function Rom_OpenISYSAllFiles ()^ On Error GoTo
NoOpenISYSAll_Err^ Dim ret As Integer^ Dim DBFileName As String^ Dim DBName As
String^ Dim ISYSFirst As Integer^ Dim Paath As String^ ' * see if there is an
ISYS database in the Rom_ISYSTextPath^ DBFileName = Rom_ISYSTextPath &
ROM_ISYS_AINDEX^ If Dir$(DBFileName) = "" Then^ MsgBox "There is no Full Text
database.", ROM_MB_OK + ROM_MB_ICONINFORMATION, ROMULUS_APP_NAME^ GoTo
ISYSAll_Exit^ End If^ ' * open ISYS or make it the active window^ ret =
Rom_OpenISYS(ROM_TR_INI_ENTRY, ROM_ISYS_WINDOWCLASS, Rom_ISYS_Windowname)^ ' *
clear the last search and make sure there is no filter set^ DBName =
Rom_ISYSTextPath^ If ret Then^ SendKeys "%fc%ff" & "{DEL}" & "{ENTER}"^
'SendKeys "%fd" & "{TAB}" & "{TAB}"^ 'SendKeys "+{END}"^ 'SendKeys "{DEL}" &
DBName & "{TAB}" & "{ENTER}"^ End If^ ' SendKeys "%fd" & "{TAB}" & "{TAB}"^
SendKeys "+{END}"^ ' If Paath = DbName Then^ ' SendKeys "{TAB}" & "{ENTER}"^
SendKeys "%fc%ff" & "{DEL}" & "{ENTER}"^ ' GoTo ISYSAll_Exit^ ' End If^ ''
If Paath <> DbName Then^ ' SendKeys "+{END}"^ ' SendKeys "{DEL}" & DbName &
"{TAB}" & "{ENTER}"^ ' SendKeys "%fc%ff" & "{DEL}" & "{ENTER}"^ ' End If^
ISYSAll_Exit:^ Exit Function^ NoOpenISYSAll_Err:^ MsgBox "Cannot open ISYS.
Error: " + Error$, ROM_MB_OK + ROM_MB_ICONEXCLAMATION, ROMULUS_APP_NAME^
Resume ISYSAll_Exit^ End Function^ Function Rom_OpenISYSFile (Filename As
String)^ On Error GoTo ISYSFile_Err^ Dim ret As Integer^ Dim FullFileName As
String^ Dim DBFileName As String^ Dim DBName As String^ ' * see if there is an
ISYS database in the Rom_ISYSTextPath^ DBFileName = Rom_ISYSTextPath &
ROM_ISYS_AINDEX^ If Dir$(DBFileName) = "" Then^ MsgBox "There is no Full Text
database.", ROM_MB_OK + ROM_MB_ICONINFORMATION, ROMULUS_APP_NAME^ GoTo
ISYSFile_Exit^ End If^ ' * see if the filename is valid^ FullFileName =
Rom_ISYSTextPath & Filename^ If Dir$(FullFileName) = "" Then^ MsgBox "There is
no associated Full Text file", ROM_MB_OK + ROM_MB_ICONINFORMATION,
ROMULUS_APP_NAME^ GoTo ISYSFile_Exit^ End If^ ' * open ISYS or make it the
active window^ ret = Rom_OpenISYS(ROM_TR_INI_ENTRY, ROM_ISYS_WINDOWCLASS,
Rom_ISYS_Windowname)^ ' * clear last search and open filename^ DBName =

```

- 44 -

```

Rom_ISYSFilePath^ If ret Then^ SendKeys "%fc%ff" & Filename & "{ENTER}"^
SendKeys "%q"^ End If^ ISYSFile_Exit:^ Exit Function^ ISYSFile_Err:^ MsgBox
"Cannot open ISYS text file. Error: " & Error$, ROM_MB_OK +
ROM_MB_ICONEXCLAMATION, ROMULUS_APP_NAME^ Resume ISYSFile_Exit^ End Function^
Function Rom_OpenLotus123 ()^ On Error GoTo NoOpenLotus123_Err^ Dim ret As
Integer^ ret = Rom_OpenApp(ROM_SPREAD2_INI_ENTRY, ROM_LOTUS123_WINDOWCLASS,
ROM_LOTUS123_WINDOWNAME)^ Lotus123_Exit:^ Exit Function^ NoOpenLotus123_Err:^
MsgBox "Cannot open Lotus 123. Error: " & Error$, ROM_MB_OK +
ROM_MB_ICONEXCLAMATION, ROMULUS_APP_NAME^ Resume Lotus123_Exit^ End Function^
Function Rom_OpenLotus123File (Filename As String)^ On Error GoTo
Lotus123File_Err^ Dim ret As Integer^ ' * see if the filename is valid^
Filename = Rom_SpreadSheetPath & Filename^ If Dir$(Filename) = "" Then^ ret =
MsgBox("There is no associated spreadsheet. Would you like to create a
spreadsheet?", ROM_MB_YESNO + MB_DEFBUTTON2 + ROM_MB_ICONINFORMATION,
ROMULUS_APP_NAME)^ If ret = 7 Then GoTo Lotus123File_Exit^ DoCmd RunMacro
"Indicator"^ ret = Rom_OpenApp(ROM_SPREAD2_INI_ENTRY,
ROM_LOTUS123_WINDOWCLASS, ROM_LOTUS123_WINDOWNAME)^ SendKeys "%fa" & Filename
& "{ENTER}"^ Exit Function^ End If^ ' * open Lotus 123 or make it the active
window^ ret = Rom_OpenApp(ROM_SPREAD2_INI_ENTRY, ROM_LOTUS123_WINDOWCLASS,
ROM_LOTUS123_WINDOWNAME)^ ' * open the filename^ If ret = True Then^ SendKeys
"%fo" & Filename & "{ENTER}"^ End If^ Lotus123File_Exit:^ Exit Function^
Lotus123File_Err:^ MsgBox "Cannot open Lotus 123 spreadsheet. Error: " &
Error$, ROM_MB_OK + ROM_MB_ICONEXCLAMATION, ROMULUS_APP_NAME^ Resume
Lotus123File_Exit^ End Function^ Function Rom_OpenOI (AppIniEntry As String,
AppClass As String, AppWindow As String) As Integer^ On Error GoTo OpenOI_Err
' Set up error handler.^ Rom_OpenOI = True^ Dim WindowHandle As Integer^ Dim
ret As Integer^ Dim lpIniRetString As String^ Dim AppName As String^ Dim
RomName As String^ Dim AppModuleName As String^ Dim ParamPos As Integer^ ' *
find the filename of the app from the ini_file^ AppName =
Rom_ReturnIniString(ROM_APP_INI_SECTION, AppIniEntry)^ If (AppName = "") Then^
MsgBox "There is no filename listed for " & AppWindow & "in the ROMULUS.INI
file.", ROM_MB_OK + ROM_MB_ICONEXCLAMATION, ROMULUS_APP_NAME^ Rom_OpenOI =
False^ GoTo OpenOI_Exit^ End If^ ' * load the app if it is not already loaded^
' * to find the module ignore any command line parameters in the name^
ParamPos = InStr(AppName, " ")^ If (ParamPos) Then^ AppModuleName =
Left$(AppName, ParamPos - 1)^ Else^ AppModuleName = AppName^ End If^ If
(Rom_GetModuleHandle(AppModuleName) = 0) Then 'App is not loaded^ ret =
Shell(AppName, 1) 'start App^ If ret = 0 Then GoTo OpenOI_Err^ ' * wait until

```

- 45 -

```

the main window of the app has been displayed^ ' ??? if it does not start up
correctly we get stuck here forever^ While (Rom_FuzzyFindWindow(AppClass,
AppWindow) = 0)^ Rom_Yield^ Wend^ End If^ ' * make sure we can find the window
handle before we open Rom link^ WindowHandle = Rom_FuzzyFindWindow(AppClass,
AppWindow)^ If WindowHandle = 0 Then^ MsgBox "Cannot Access " & AppWindow,
ROM_MB_OK + ROM_MB_ICONEXCLAMATION, ROMULUS_APP_NAME^ Rom_OpenOI = False^ GoTo
OpenOI_Exit^ End If^ ' * Start the Romulus floating top button window^ ' * If
it doesn't start we still let them go ahead^ RomName =
Rom_ReturnIniString(ROM_APP_INI_SECTION, ROMLINK_INI_ENTRY)^ RomName = RomName
& " " & AppWindow^ If (RomName = "") Then^ MsgBox "There is no filename listed
for the Romulus link program in the ROMULUS.INI file.", ROM_MB_OK +
ROM_MB_ICONEXCLAMATION, ROMULUS_APP_NAME^ Else^ ret = Shell(RomName, 8) 'start
Romulus floating top window.^ End If^ ' * maximize the window and make it the
active window^ If (Rom_IsZoomed(WindowHandle) = 0) Then^ ' ret =
Rom_ShowWindow(WindowHandle, ROM_SW_SHOWMAXIMIZED)^ 'Else^ ret =
Rom_SetActiveWindow(WindowHandle)^ End If^ OpenOI_Exit:^ Exit Function^
OpenOI_Err:^ MsgBox "Cannot open " & AppWindow & ". Error: " + Error$,
ROM_MB_OK + ROM_MB_ICONEXCLAMATION, ROMULUS_APP_NAME^ Rom_OpenOI = False^
Resume OpenOI_Exit^ End Function^ Function Rom_OpenOIDFile (Filename As
String)^ On Error GoTo OIDFile_Err^ Dim ret As Integer^ SendKeys "%fs" &
Filename & "{ENTER}"^ ' * open OPEN/image or make it the active window^ ret =
Rom_OpenApp(ROM_VIEWER2_INI_ENTRY, ROM_OPENIMAGE_WINDOWCLASS,
ROM_OPENIMAGE_WINDOWNAME)^ ' * open the filename^ If ret = True Then^ SendKeys
"%do" & Filename & "{ENTER}"^ End If^ OIDFile_Exit:^ Exit Function^
OIDFile_Err:^ MsgBox "Cannot open OPEN/image image document. Error: " +
Error$, ROM_MB_OK + ROM_MB_ICONEXCLAMATION, ROMULUS_APP_NAME^ Resume
OIDFile_Exit^ End Function^ Function Rom_OpenOIFFile (Filename As String)^ On
Error GoTo OIFile_Err^ Dim ret As Integer^ ' * see if the filename is valid^
Filename = Rom_ImagePath & Filename^ If Dir$(Filename) = "" Then^ ret =
MsgBox("There is no associated image. Would you like to create an image?",
ROM_MB_YESNO + MB_DEFBUTTON2 + ROM_MB_ICONQUESTION, ROMULUS_APP_NAME)^ If ret
= 7 Then GoTo OIFile_Exit^ DoCmd RunMacro "Indicator"^ ret =
Rom_OpenApp(ROM_VIEWER2_INI_ENTRY, ROM_OPENIMAGE_WINDOWCLASS,
ROM_OPENIMAGE_WINDOWNAME)^ ' * ret = Rom_OpenApp(ROM_VIEWER3_INI_ENTRY,
ROM_FTCOLOR_WINDOWCLASS, ROM_FTCOLOR_WINDOWNAME)^ ' * SendKeys "%fnl" &
"{ENTER}"^ ' * SendKeys "%fa" & Filename & "{ENTER}"^ ' * SendKeys "%fq"^ Exit
Function^ End If^ SendKeys "%fs" & Filename & "{ENTER}"^ ' * open OPEN/image
or make it the active window^ ret = Rom_OpenApp(ROM_VIEWER2_INI_ENTRY,

```

- 46 -

```

ROM_OPENIMAGE_WINDOWCLASS, ROM_OPENIMAGE_WINDOWNAME)^ ' * open the filename^
If ret = True Then^ SendKeys "%fo" & Filename & "{ENTER}"^ End If^
OIFFFile_Exit:^ Exit Function^ OIFFFile_Err:^ MsgBox "Cannot open OPEN/image
image document. Error: " & Error$, ROM_MB_OK + ROM_MB_ICONEXCLAMATION,
ROMULUS_APP_NAME^ Resume OIFFFile_Exit^ End Function^ Function
Rom_OpenOpenImage ()^ On Error GoTo NoOpenOI_Err^ Dim ret As Integer^ ret =
Rom_OpenApp(ROM_VIEWER2_INI_ENTRY, ROM_OPENIMAGE_WINDOWCLASS,
ROM_OPENIMAGE_WINDOWNAME)^ OI_Exit:^ Exit Function^ NoOpenOI_Err:^ MsgBox
"Cannot open OPEN/image. Error: " & Error$, ROM_MB_OK +
ROM_MB_ICONEXCLAMATION, ROMULUS_APP_NAME^ Resume OI_Exit^ End Function^
Function Rom_OpenORG ()^ On Error GoTo NoOpenORG_Err^ Dim ret As Integer^ ret =
Rom_OpenApp(ROM_ORG_INI_ENTRY, ROM_ORG_WINDOWCLASS, ROM_ORG_WINDOWNAME)^
ORG_Exit:^ Exit Function^ NoOpenORG_Err:^ MsgBox "Cannot open Organizer.
Error: " & Error$, ROM_MB_OK + ROM_MB_ICONEXCLAMATION, ROMULUS_APP_NAME^
Resume ORG_Exit^ End Function^ Function Rom_OpenPStyler ()^ On Error GoTo
NoOpenPStyler_Err^ Dim ret As Integer^ ret =
Rom_OpenApp(ROM_VIEWER2_INI_ENTRY, ROM_PSTYLER_WINDOWCLASS,
ROM_PSTYLER_WINDOWNAME)^ PStyler_Exit:^ Exit Function^ NoOpenPStyler_Err:^
MsgBox "Cannot open Photo Styler. Error: " & Error$, ROM_MB_OK +
ROM_MB_ICONEXCLAMATION, ROMULUS_APP_NAME^ Resume PStyler_Exit^ End Function^
Function Rom_OpenPStylerFile (Filename As String)^ On Error GoTo
PStylerFile_Err^ Dim ret As Integer^ ' * see if the filename is valid^
Filename = Rom_ImagePath & Filename^ If Dir$(Filename) = "" Then^ ret =
MsgBox("There is no associated image. Would you like to create an image?",
ROM_MB_YESNO + MB_DEFBUTTON2 + ROM_MB_ICONINFORMATION, ROMULUS_APP_NAME)^ If
ret = 7 Then GoTo PStylerFile_Exit^ DoCmd RunMacro "Indicator"^ ret =
Rom_OpenApp(ROM_VIEWER3_INI_ENTRY, ROM_FTCOLOR_WINDOWCLASS,
ROM_FTCOLOR_WINDOWNAME)^ SendKeys "%fnil" & "{ENTER}"^ SendKeys "%fa" &
Filename & "{ENTER}"^ SendKeys "%fq"^ Exit Function^ GoTo PStylerFile_Exit^
End If^ ' * open PStyler or make it the active window^ ret =
Rom_OpenApp(ROM_VIEWER2_INI_ENTRY, ROM_PSTYLER_WINDOWCLASS,
ROM_PSTYLER_WINDOWNAME)^ ' * open the filename^ If ret = True Then^ SendKeys
"%fo" & Filename & "{ENTER}"^ End If^ PStylerFile_Exit:^ Exit Function^
PStylerFile_Err:^ MsgBox "Cannot open Photo Styler image. Error: " & Error$,
ROM_MB_OK + ROM_MB_ICONEXCLAMATION, ROMULUS_APP_NAME^ Resume PStylerFile_Exit^
End Function^ Function Rom_OpenQpro ()^ On Error GoTo NoOpenQpro_Err^ Dim ret
As Integer^ ret = Rom_OpenApp(ROM_SPREAD3_INI_ENTRY, ROM_QPRO_WINDOWCLASS,
ROM_QPRO_WINDOWNAME)^ Qpro_Exit:^ Exit Function^ NoOpenQpro_Err:^ MsgBox

```

- 47 -

```

"Cannot open Quattro Pro. Error: " + Error$, ROM_MB_OK +
ROM_MB_ICONEXCLAMATION, ROMULUS_APP_NAME^ Resume Qpro_Exit^ End Function^
Function Rom_OpenQproFile (Filename As String)^ On Error GoTo QproFile_Err^
Dim ret As Integer^ ' * see if the filename is valid^ Filename =
Rom_SpreadSheetPath & Filename^ If Dir$(Filename) = "" Then^ ret =
MsgBox("There is no associated spreadsheet. Would you like to create a
spreadsheet?", ROM_MB_YESNO + MB_DEFBUTTON2 + ROM_MB_ICONINFORMATION,
ROMULUS_APP_NAME)^ If ret = 7 Then GoTo QproFile_Exit^ DoCmd RunMacro
"Indicator"^ ret = Rom_OpenApp(ROM_SPREAD3_INI_ENTRY, ROM_QPRO_WINDOWCLASS,
ROM_QPRO_WINDOWNAME)^ SendKeys "%fa" & Filename & "{ENTER}"^ Exit Function^
End If^ ' * open Quattro Pro or make it the active window^ ret =
Rom_OpenApp(ROM_SPREAD3_INI_ENTRY, ROM_QPRO_WINDOWCLASS, ROM_QPRO_WINDOWNAME)^
' * open the filename^ If ret = True Then^ SendKeys "%fo" & Filename &
"{ENTER}"^ End If^ QproFile_Exit:^ Exit Function^ QproFile_Err:^ MsgBox
"Cannot open Quattro Pro spreadsheet. Error: " + Error$, ROM_MB_OK +
ROM_MB_ICONEXCLAMATION, ROMULUS_APP_NAME^ Resume QproFile_Exit^ End Function^
Function Rom_OpenWord ()^ On Error GoTo NoOpenWord_Err^ Dim ret As Integer^
ret = Rom_OpenApp(ROM_WP_INI_ENTRY, ROM_WORD_WINDOWCLASS,
ROM_WORD_WINDOWNAME)^ Word_Exit:^ Exit Function^ NoOpenWord_Err:^ MsgBox
"Cannot open Word for Windows. Error: " + Error$, ROM_MB_OK +
ROM_MB_ICONEXCLAMATION, ROMULUS_APP_NAME^ Resume Word_Exit^ End Function^
Function Rom_OpenWord6File (Filename As String)^ On Error GoTo Word6File_Err^
Dim ret As Integer^ ' * see if the filename is valid^ Filename =
Rom_DocumentPath & Filename^ If Dir$(Filename) = "" Then^ ret = MsgBox("There
is no associated document. Would you like to create a document?", ROM_MB_YESNO
+ MB_DEFBUTTON2 + ROM_MB_ICONINFORMATION, ROMULUS_APP_NAME)^ If ret = 7 Then
GoTo Word6File_Exit^ DoCmd RunMacro "Indicator"^ ret =
Rom_OpenApp(ROM_WP_INI_ENTRY, ROM_WORD_WINDOWCLASS, ROM_WORD_WINDOWNAME)^
SendKeys "%fn" & "{ENTER}"^ SendKeys "%fa" & Filename & "{ENTER}"^ Exit
Function^ End If^ ' * open Word or make it the active window^ ret =
Rom_OpenApp(ROM_WP_INI_ENTRY, ROM_WORD_WINDOWCLASS, ROM_WORD_WINDOWNAME)^ ' *
open the filename^ If ret = True Then^ SendKeys "%fo" & Filename & "{ENTER}"^
End If^ Word6File_Exit:^ Exit Function^ Word6File_Err:^ MsgBox "Cannot open
Word for Windows document. Error: " + Error$, ROM_MB_OK +
ROM_MB_ICONEXCLAMATION, ROMULUS_APP_NAME^ Resume Word6File_Exit^ End Function^
Function Rom_OpenWordFile (Docnum As String)^ On Error GoTo WordFile_Err^ Dim
ret As Integer^ Dim Imagename As String^ Dim Filename As String^ ' * see if
the filename is valid^ Filename = Rom_DocumentPath & Docnum & ".doc"^

```

- 48 -

```

Imagename = Rom_ImagePath & Docnum & ".tif"
If Dir$(Filename) = "" Then ret =
MsgBox("There is no associated document. Would you like to create a
document?", ROM_MB_YESNO + MB_DEFBUTTON2 + ROM_MB_ICONQUESTION,
ROMULUS_APP_NAME)
If ret = 7 Then GoTo WordFile_Exit
DoCmd RunMacro
"Indicator"
If Dir$(Imagename) <> "" Then Else GoTo WordFile_Next
ret =
MsgBox("Would you like to OCR the related image?", ROM_MB_YESNO +
MB_DEFBUTTON2 + ROM_MB_ICONQUESTION, ROMULUS_APP_NAME)
If ret = 7 Then GoTo
WordFile_Next
ret = Rom_OpenApp(ROM_WP_INI_ENTRY, ROM_WORD_WINDOWCLASS,
ROM_WORD_WINDOWNAME)
SendKeys "%fn" & "{ENTER}"
SendKeys "%fa" & Filename &
"{ENTER}"
SendKeys "%fq", True
SendKeys "{RIGHT 500}", True
SendKeys "{LEFT 500}", True
SendKeys "{RIGHT 500}", True
SendKeys "{LEFT 500}", True
SendKeys "%d" & "{ENTER}"
SendKeys Imagename & "{ENTER}"
Exit Function
End
If ' * open Word or make it the active window
ret =
Rom_OpenApp(ROM_WP_INI_ENTRY, ROM_WORD_WINDOWCLASS, ROM_WORD_WINDOWNAME)
' *
open the filename
If ret = True Then SendKeys "%fo" & Filename & "{ENTER}"
Exit Function
End
If WordFile_Next: ret = Rom_OpenApp(ROM_WP_INI_ENTRY,
ROM_WORD_WINDOWCLASS, ROM_WORD_WINDOWNAME)
SendKeys "%fn" & "{ENTER}"
SendKeys "%fa" & Filename & "{ENTER}"
Exit Function
WordFile_Exit: Exit
Function
WordFile_Err: MsgBox "Cannot open Word for Windows document. Error:
" + Error$, ROM_MB_OK + ROM_MB_ICONEXCLAMATION, ROMULUS_APP_NAME
Resume
WordFile_Exit
End Function
Function Rom_OpenWordPerfect ()
On Error GoTo
NoOpenWordPerfect_Err
Dim ret As Integer
ret =
Rom_OpenApp(ROM_WP2_INI_ENTRY, ROM_WORDPERFECT_WINDOWCLASS,
ROM_WORDPERFECT_WINDOWNAME)
WordPerfect_Exit: Exit Function
NoOpenWordPerfect_Err: MsgBox "Cannot open Word Perfect. Error: " + Error$,
ROM_MB_OK + ROM_MB_ICONEXCLAMATION, ROMULUS_APP_NAME
Resume WordPerfect_Exit
End Function
Function Rom_OpenWordPerfectFile (Filename As String)
On Error
GoTo WPFile_Err
Dim ret As Integer
' * see if the filename is valid
Filename = Rom_DocumentPath & Filename
If Dir$(Filename) = "" Then ret =
MsgBox("There is no associated document. Would you like to create a
document?", ROM_MB_YESNO + MB_DEFBUTTON2 + ROM_MB_ICONINFORMATION,
ROMULUS_APP_NAME)
If ret = 7 Then GoTo WPFile_Exit
DoCmd RunMacro
"Indicator"
ret = Rom_OpenApp(ROM_WP2_INI_ENTRY, ROM_WORDPERFECT_WINDOWCLASS,
ROM_WORDPERFECT_WINDOWNAME)
While
(Rom_FuzzyFindWindow(ROM_WORDPERFECT_WINDOWCLASS, ROM_WORDPERFECT_WINDOWNAME)
= 0)
Rom_Yield
Wend
SendKeys "%fn" & "{ENTER}"
SendKeys "%fa" & Filename &
"{ENTER}"
Exit Function
End
If ' * open Word or make it the active window
ret = Rom_OpenApp(ROM_WP2_INI_ENTRY, ROM_WORDPERFECT_WINDOWCLASS,

```

- 49 -

```

ROM_WORDPERFECT_WINDOWNAME)^ ' * open the filename^ If ret = True Then^
SendKeys "%fo" & Filename & "{ENTER}"^ End If^ WPFile_Exit:^ Exit Function^
WPFile_Err:^ MsgBox "Cannot open Word Perfect document. Error: " + Error$,
ROM_MB_OK + ROM_MB_ICONEXCLAMATION, ROMULUS_APP_NAME^ Resume WPFile_Exit^ End
Function^ Function Rom_OpenWordScan ()^ On Error GoTo NoOpenWSC_Err^ Dim ret.
As Integer^ ret = Rom_OpenApp(ROM_VIEWER_INI_ENTRY, ROM_WORDSCAN_WINDOWCLASS,
ROM_WORDSCAN_WINDOWNAME)^ WSC_Exit:^ Exit Function^ NoOpenWSC_Err:^ MsgBox
"Cannot open Word Scan. Error: " + Error$, ROM_MB_OK + ROM_MB_ICONEXCLAMATION,
ROMULUS_APP_NAME^ Resume WSC_Exit^ End Function^ Function Rom_OpenWordScanFile
(Filename As String)^ On Error GoTo WSCFile_Err^ Dim ret As Integer^ ' * see
if the filename is valid^ Filename = Rom_ImagePath & Filename^ If
Dir$(Filename) = "" Then^ ret = MsgBox("There is no associated image. Would
you like to create an image?", ROM_MB_YESNO + MB_DEFBUTTON2 +
ROM_MB_ICONINFORMATION, ROMULUS_APP_NAME)^ If ret = 7 Then GoTo WSCFile_Exit^
DoCmd RunMacro "Indicator"^ ret = Rom_OpenApp(ROM_VIEWER3_INI_ENTRY,
ROM_FTCOLOR_WINDOWCLASS, ROM_FTCOLOR_WINDOWNAME)^ SendKeys "%fnil" &
"{ENTER}"^ SendKeys "%fa" & Filename & "{ENTER}"^ SendKeys "%fq"^ Exit
Function^ GoTo WSCFile_Exit^ End If^ ' * open Word Scan or make it the active
window^ ret = Rom_OpenApp(ROM_VIEWER_INI_ENTRY, ROM_WORDSCAN_WINDOWCLASS,
ROM_WORDSCAN_WINDOWNAME)^ ' * open the filename^ If ret = True Then^ SendKeys
"%fnd" & Filename & "{ENTER}"^ End If^ WSCFile_Exit:^ Exit Function^
WSCFile_Err:^ MsgBox "Cannot open Word Scan image. Error: " + Error$,
ROM_MB_OK + ROM_MB_ICONEXCLAMATION, ROMULUS_APP_NAME^ Resume WSCFile_Exit^ End
Function^ '** * Romulus - Path Module^ '* Copyright 1993 Romulus Software^
'** Option Compare Database^ 'Use database order for string comparisons^ Option
Explicit^ Function Rom_SavePaths ()^ On Error GoTo SavePath_Err^ Dim DB As
Database^ Dim PathsDyna As Dynaset^ Dim ret As Integer^ Set DB = CurrentDB(^
Set PathsDyna = DB.CreateDynaset("Romulus Paths")^ PathsDyna.Edit^
Rom_DocumentPath = forms("Romulus Paths Form").form("DocPath")^ If
Right$(Rom_DocumentPath, 1) <> "\" Then Rom_DocumentPath = Rom_DocumentPath +
"\^ PathsDyna("Document Path") = Rom_DocumentPath^ Rom_ImagePath =
forms("Romulus Paths Form").form("ImagePath")^ If Right$(Rom_ImagePath, 1) <>
"\ Then Rom_ImagePath = Rom_ImagePath + "\^ PathsDyna("Image Path") =
Rom_ImagePath^ Rom_ISYSTextPath = forms("Romulus Paths
Form").form("TextPath")^ If Right$(Rom_ISYSTextPath, 1) <> "\" Then
Rom_ISYSTextPath = Rom_ISYSTextPath + "\^ PathsDyna("ISYSText Path") =
Rom_ISYSTextPath^ Rom_SpreadSheetPath = forms("Romulus Paths
Form").form("SpreadPath")^ If Right$(Rom_SpreadSheetPath, 1) <> "\" Then

```

- 50 -

```

Rom_SpreadSheetPath = Rom_SpreadSheetPath + "\"^ PathsDyna("Spreadsheet Path")
= Rom_SpreadSheetPath^ Rom_ChartPath = forms("Romulus Paths
Form").form("ChartPath")^ If Right$(Rom_ChartPath, 1) <> "\" Then
Rom_ChartPath = Rom_ChartPath + "\"^ PathsDyna("Chart Path") = Rom_ChartPath^
PathsDyna.Update^ DB.Close^ PathsDyna.Close^ ret = Rom_ClosePathForm()^
SavePath_Exit:^ Exit Function^ SavePath_Err:^ MsgBox "Error: " + Error$,
ROM_MB_OK + ROM_MB_ICONEXCLAMATION, ROMULUS_APP_NAME^ Resume SavePath_Exit^
End Function^ Function Rom_InitPathForm ()^ On Error GoTo InitPath_Err^ Dim DB
As Database^ Dim PathsDyna As Dynaset^ Set DB = CurrentDB()^ Set PathsDyna =
DB.CreateDynaset("Romulus Paths")^ forms("Romulus Paths Form").form("DocPath")
= PathsDyna("Document Path")^ forms("Romulus Paths Form").form("ImagePath") =
PathsDyna("Image Path")^ forms("Romulus Paths Form").form("TextPath") =
PathsDyna("ISYSText Path")^ forms("Romulus Paths Form").form("SpreadPath") =
PathsDyna("Spreadsheet Path")^ forms("Romulus Paths Form").form("ChartPath") =
PathsDyna("Chart Path")^ DB.Close^ PathsDyna.Close^ InitPath_Exit:^ Exit
Function^ InitPath_Err:^ MsgBox "Error: " + Error$, ROM_MB_OK +
ROM_MB_ICONEXCLAMATION, ROMULUS_APP_NAME^ Resume InitPath_Exit^ End Function^
^ '* Romulus - Utilities Module^ '* Copyright 1993 Romulus Software^ '*
Option Compare Database 'Use database order for string comparisons^ Option
Explicit^ Function Rom_Backup ()^ On Error GoTo Backup_Err^ Dim ret As
Integer^ Dim FileName As String^ Dim RomName As String^ ' * find the filename
of the app from the ini_file^ FileName =
Rom_ReturnIniString(ROM_APP_INI_SECTION, ROM_BACKUP_INI_ENTRY)^ If (FileName =
"") Then^ MsgBox "There is no filename listed for the Backup program in the
ROMULUS.INI file.", ROM_MB_OK + ROM_MB_ICONEXCLAMATION, ROMULUS_APP_NAME^ GoTo
Backup_Exit^ End If^ ' * open Backup or make it the active window^ ret =
Shell(FileName, 1)^ Backup_Exit:^ Exit Function^ Backup_Err:^ MsgBox "Error: "
+ Error$, ROM_MB_OK + ROM_MB_ICONEXCLAMATION, ROMULUS_APP_NAME^ Resume
Backup_Exit^ End Function^ Function Rom_Word6Convert ()^ On Error GoTo
Word6Convert_Err^ Dim ret As Integer^ Dim FileName As String^ ' * find the
convert template name from the ini_file^ FileName =
Rom_ReturnIniString(ROM_APP_INI_SECTION, ROM_CONVERT_INI_ENTRY)^ If (FileName
= "") Then^ MsgBox "The Convert template is not listed in the ROMULUS.INI
file.", ROM_MB_OK + ROM_MB_ICONEXCLAMATION, ROMULUS_APP_NAME^ GoTo
Word6Convert_Exit^ End If^ ' * open Word or make it the active window^ ret =
Rom_OpenApp(ROM_WP_INI_ENTRY, ROM_WORD_WINDOWCLASS, ROM_WORD_WINDOWNAME)^ ' *
open the filename^ If ret = True Then^ SendKeys "%fn" & FileName & "{ENTER}"^
End If^ Word6Convert_Exit:^ Exit Function^ Word6Convert_Err:^ MsgBox "Error: "

```

- 51 -

```

+ Error$, ROM_MB_OK + ROM_MB_ICONEXCLAMATION, ROMULUS_APP_NAME^ Resume
Word6Convert_Exit^ End Function^ Function Rom_Word6ISYSCapture ()^ On Error
GoTo Word6Capture_Err^ Dim ret As Integer^ Dim FileName As String^ ' * find
the text edit template name from the ini_file^ FileName =
Rom_ReturnIniString(ROM_APP_INI_SECTION, ROM_TEXTSAVE_INI_ENTRY)^ If (FileName
= "") Then^ MsgBox "The ISYS Capture template is not listed in the ROMULUS.INI
file.", ROM_MB_OK + ROM_MB_ICONEXCLAMATION, ROMULUS_APP_NAME^ GoTo
Word6Capture_Exit^ End If^ ' * open Word or make it the active window^ ret =
Rom_OpenApp(ROM_WP_INI_ENTRY, ROM_WORD_WINDOWCLASS, ROM_WORD_WINDOWNAME)^ ' *
open the filename^ If ret = True Then^ SendKeys "%fn" & FileName & "{ENTER}"^
SendKeys "%fq"^ End If^ Word6Capture_Exit:^ Exit Function^ Word6Capture_Err:^
MsgBox "Error: " + Error$, ROM_MB_OK + ROM_MB_ICONEXCLAMATION,
ROMULUS_APP_NAME^ Resume Word6Capture_Exit^ End Function^ Function
Rom_Word6Join ()^ On Error GoTo Word6Join_Err^ Dim ret As Integer^ Dim
FileName As String^ ' shut the utilities form first so that we don't lock up
when romlink sets Access as the active window^ DoCmd Close A_FORM, "Romulus
Utilities"^ ' * find the join template name from the ini_file^ FileName =
Rom_ReturnIniString(ROM_APP_INI_SECTION, ROM_JOIN_INI_ENTRY)^ If (FileName =
 "") Then^ MsgBox "The Join template is not listed in the ROMULUS.INI file.",
ROM_MB_OK + ROM_MB_ICONEXCLAMATION, ROMULUS_APP_NAME^ GoTo Word6Join_Exit^ End
If^ ' * open Word or make it the active window^ ret =
Rom_OpenApp(ROM_WP_INI_ENTRY, ROM_WORD_WINDOWCLASS, ROM_WORD_WINDOWNAME)^ ' *
open the filename^ If ret = True Then^ SendKeys "%fn" & FileName & "{ENTER}"^
End If^ Word6Join_Exit:^ Exit Function^ Word6Join_Err:^ MsgBox "Error: " +
Error$, ROM_MB_OK + ROM_MB_ICONEXCLAMATION, ROMULUS_APP_NAME^ Resume
Word6Join_Exit^ End Function^ Function Rom_WordOCRCapture ()^ On Error GoTo
WordOCR_Err^ Dim ret As Integer^ ret = Rom_OpenApp(ROM_WP_INI_ENTRY,
ROM_WORD_WINDOWCLASS, ROM_WORD_WINDOWNAME)^ SendKeys "%fn{ENTER}"^
WordOCR_Exit:^ Exit Function^ WordOCR_Err:^ MsgBox "Error: " + Error$,
ROM_MB_OK + ROM_MB_ICONEXCLAMATION, ROMULUS_APP_NAME^ Resume WordOCR_Exit^ End
Function^ '** * Romulus - Menus Module^ * Copyright 1993 Romulus Software^
** Option Compare Database 'Use database order for string comparisons^ Option
Explicit^ Function Rom_Exit ()^ On Error GoTo RomExit_Err^ Dim hWnd As
Integer^ Dim ret As Integer^ ' * confirm they really want to exit^ ret =
MsgBox("Are you sure you wish to exit?" + Error$, ROM_MB_YESNO + MB_DEFBUTTON2
+ ROM_MB_ICONQUESTION, ROMULUS_APP_NAME)^ If (ret = 7) Or (ret = 2) Then GoTo
RomExit_Exit^ hWnd = Rom_FuzzyFindWindow(ROM_WORD_WINDOWCLASS,
ROM_WORD_WINDOWNAME)^ If hWnd Then^ ret = Rom_PostMessage(hWnd, ROM_WM_CLOSE,

```

- 52 -

```

0, 0)^ End If^ hWnd = Rom_FuzzyFindWindow(ROM_EXCEL_WINDOWCLASS,
ROM_EXCEL_WINDOWNAME)^ If hWnd Then^ ret = Rom_PostMessage(hWnd, ROM_WM_CLOSE,
0, 0)^ End If^ hWnd = Rom_FuzzyFindWindow(ROM_ISYS_WINDOWCLASS,
ROM_ISYS_WINDOWNAME)^ If hWnd Then^ ret = Rom_PostMessage(hWnd, ROM_WM_CLOSE,
0, 0)^ End If^ hWnd = Rom_FuzzyFindWindow(ROM_PSTYLER_WINDOWCLASS,
ROM_PSTYLER_WINDOWNAME)^ If hWnd Then^ ret = Rom_PostMessage(hWnd,
ROM_WM_CLOSE, 0, 0)^ End If^ hWnd =
Rom_FuzzyFindWindow(ROM_FTCOLOR_WINDOWCLASS, ROM_FTCOLOR_WINDOWNAME)^ If hWnd
Then^ ret = Rom_PostMessage(hWnd, ROM_WM_CLOSE, 0, 0)^ End If^ hWnd =
Rom_FuzzyFindWindow(ROM_ABC_WINDOWCLASS, ROM_ABC_WINDOWNAME)^ If hWnd Then^
ret = Rom_PostMessage(hWnd, ROM_WM_CLOSE, 0, 0)^ End If^ hWnd =
Rom_FuzzyFindWindow(ROM_WORDPERFECT_WINDOWCLASS, ROM_WORDPERFECT_WINDOWNAME)^
If hWnd Then^ ret = Rom_PostMessage(hWnd, ROM_WM_CLOSE, 0, 0)^ End If^ hWnd =
Rom_FuzzyFindWindow(ROM_AMIPRO_WINDOWCLASS, ROM_AMIPRO_WINDOWNAME)^ If hWnd
Then^ ret = Rom_PostMessage(hWnd, ROM_WM_CLOSE, 0, 0)^ End If^ hWnd =
Rom_FuzzyFindWindow(ROM_LOTUS123_WINDOWCLASS, ROM_LOTUS123_WINDOWNAME)^ If
hWnd Then^ ret = Rom_PostMessage(hWnd, ROM_WM_CLOSE, 0, 0)^ End If^ hWnd =
Rom_FuzzyFindWindow(ROM_QPRO_WINDOWCLASS, ROM_QPRO_WINDOWNAME)^ If hWnd Then^
ret = Rom_PostMessage(hWnd, ROM_WM_CLOSE, 0, 0)^ End If^ hWnd =
Rom_FuzzyFindWindow(ROM_WORDSCAN_WINDOWCLASS, ROM_WORDSCAN_WINDOWNAME)^ If
hWnd Then^ ret = Rom_PostMessage(hWnd, ROM_WM_CLOSE, 0, 0)^ End If^ hWnd =
Rom_FuzzyFindWindow(ROM_OPENIMAGE_WINDOWCLASS, ROM_OPENIMAGE_WINDOWNAME)^ If
hWnd Then^ ret = Rom_PostMessage(hWnd, ROM_WM_CLOSE, 0, 0)^ End If^ ' *?????
label package and audio and voice and resource calc and isys update^ ' * shut
down Access^ DoCmd Quit A_PROMPT^ RomExit_Exit:^ Exit Function^ RomExit_Err:^
MsgBox "Error: " + Error$, ROM_MB_OK + ROM_MB_ICONEXCLAMATION,
ROMULUS_APP_NAME^ Resume RomExit_Exit^ End Function^' * Romulus - Numbering
Module Copyright 1993 Romulus Software^' * Option Compare Database 'Use database
order for string comparisons^Option Explicit^' * set module variable^Dim
gRom_NumsDirty As Integer^Function Rom_AddNumbers ()^On Error GoTo
AddNums_Err^Rom_AddNumbers = False^Dim Prefix As String^Dim StartNum As
String^Dim EndNum As String^Dim StartExt As String^Dim EndExt As String^Dim
PrefixVar As Variant^Dim StartNumVar As Variant^Dim EndNumVar As Variant^Dim
StartExtVar As Variant^Dim EndExtVar As Variant^Dim PrefixLen As Integer^Dim
StartNumLen As Integer^Dim EndNumLen As Integer^Dim StartExtLen As Integer^Dim
EndExtLen As Integer^Dim Num As Integer^Dim Ext As Integer^Dim Buffer As
String^Dim FileName As String^Dim FormatLine As String^Dim MaxWidth As
Integer^' * get the values of the text controls^PrefixVar =

```

- 53 -

```

UCase(forms("Romulus Numbering").form("Prefix"))^If IsNull(PrefixVar)
Then^Prefix = ""^PrefixLen = 0^Else^Prefix = PrefixVar^PrefixLen =
Len(PrefixVar)^End If^StartNumVar = forms("Romulus Numbering").form("Starting
Number")^If IsNull(StartNumVar) Then^StartNum = ""^StartNumLen =
0^Else^StartNum = StartNumVar^StartNumLen = Len(StartNumVar)^End If^EndNumVar
= forms("Romulus Numbering").form("Ending Number")^If IsNull(EndNumVar)
Then^EndNum = ""^EndNumLen = 0^Else^EndNum = EndNumVar^EndNumLen =
Len(EndNumVar)^End If^StartExtVar = UCase(forms("Romulus
Numbering").form("From Extension"))^If IsNull(StartExtVar) Then^StartExt =
""^StartExtLen = 0^Else^StartExt = StartExtVar^StartExtLen =
Len(StartExtVar)^End If^EndExtVar = UCase(forms("Romulus Numbering").form("To
Extension"))^If IsNull(EndExtVar) Then^EndExt = ""^EndExtLen = 0^Else^EndExt =
EndExtVar^EndExtLen = Len(EndExtVar)^End If^
* validate the start and end
strings^If (IsNumeric(StartNum) = False) Or (Val(StartNum) < 1) Then^MsgBox
"The 'Starting Number' must be a positive number (with optional leading
zeroes)", ROM_MB_OK + ROM_MB_ICONEXCLAMATION, ROMULUS_APP_NAME^DoCmd
GoToControl "Starting Number"^GoTo AddNums_Exit^End If^If (EndNum <> "") And
((IsNumeric(EndNum) = False) Or (Val(EndNum) < 1)) Then^MsgBox "The 'Ending
Number' must be a positive Number (with optional leading zeroes)", ROM_MB_OK +
ROM_MB_ICONEXCLAMATION, ROMULUS_APP_NAME^DoCmd GoToControl "Starting
Number"^GoTo AddNums_Exit^End If^
* validate that startnum is greater than
end num if they are both there^If (EndNum <> "") And (Val(EndNum) <
Val(StartNum)) Then^MsgBox "The 'Ending Number' must must be greater than the
'Starting Number'.", ROM_MB_OK + ROM_MB_ICONEXCLAMATION,
ROMULUS_APP_NAME^DoCmd GoToControl "Ending Number"^GoTo AddNums_Exit^End If^
* validate total lengths^If ((StartNumLen + PrefixLen) > 7) Or ((EndNumLen +
PrefixLen) > 7) Then^MsgBox "The total number of digits in the Prefix and the
number cannot exceed 7.", ROM_MB_OK + ROM_MB_ICONEXCLAMATION,
ROMULUS_APP_NAME^DoCmd GoToControl "Prefix"^GoTo AddNums_Exit^End If^
*
validate the extensions^If StartExtLen > 1 Or (StartExt <> "" And (StartExt <
"A" Or StartExt > "Z")) Then^MsgBox "The 'From Extension' must be a single
letter between 'A' and 'Z'.", ROM_MB_OK + ROM_MB_ICONEXCLAMATION,
ROMULUS_APP_NAME^DoCmd GoToControl "From Extension"^GoTo AddNums_Exit^End
If^If EndExtLen > 1 Or (EndExt <> "" And (EndExt < "A" Or EndExt > "Z"))
Then^MsgBox "The 'To Extension' must be a single letter between 'A' and 'Z'.",
ROM_MB_OK + ROM_MB_ICONEXCLAMATION, ROMULUS_APP_NAME^DoCmd GoToControl "To
Extension"^GoTo AddNums_Exit^End If^If EndExt <> "" And StartExt = ""
Then^MsgBox "There must be a 'From Extension' if you enter a 'To Extension'.",

```

- 54 -

```

ROM_MB_OK + ROM_MB_ICONEXCLAMATION, ROMULUS_APP_NAME^DoCmd GoToControl "From
Extension"^GoTo AddNums_Exit^End If^If (EndExt <> "") And (EndExt < StartExt)
Then^MsgBox "The 'To Extension' must must be a letter that occurs after the
'From Extension'.", ROM_MB_OK + ROM_MB_ICONEXCLAMATION, ROMULUS_APP_NAME^DoCmd
GoToControl "To Extension"^GoTo AddNums_Exit^End If^' * if the end number is
blank set it equal to the start^If EndNum = "" Then EndNum = StartNum^If
EndExt = "" Then EndExt = StartExt^' * determine the number of leading zeroes
(if any)^If Left$(StartNum, 1) = "0" Or Left$(EndNum, 1) = "0" Then^If
StartNumLen > EndNumLen Then^MaxWidth = StartNumLen^Else^MaxWidth =
EndNumLen^End If^Else^MaxWidth = 1^End If^FormatLine = String$(MaxWidth,
"0")^' * show hourglass^DoCmd Hourglass True^' * add the requested numbers to
the file^FileName = Rom_ReturnIniString(ROM_NUMFILE_INI_SECTION,
ROM_NUMFILE_INI_ENTRY)^Open FileName For Append As #1^If StartExt = ""
Then^For Num = Val(StartNum) To Val(EndNum)^Print #1, Chr$(34) & Prefix &
Format$(Num, FormatLine) & Chr$(34)^Next^Else^For Num = Val(StartNum) To
Val(EndNum)^For Ext = Asc(StartExt) To Asc(EndExt)^Print #1, Chr$(34) & Prefix
& Format$(Num, FormatLine) & Chr$(Ext); Chr$(34)^Next^Next^End If^' * set
module variable to indicate we are now dirty^gRom_NumsDirty = True^' * make
the print and view buttons active^forms("Romulus
Numbering").form("Disabled_ViewNumbers").Visible = False^forms("Romulus
Numbering").form("Disabled_PrintNumbers").Visible = False^forms("Romulus
Numbering").form("View Numbers").Enabled = True^forms("Romulus
Numbering").form("Print Numbers").Enabled = True^' * clear all the edit
controls^forms("Romulus Numbering").form("Prefix") = Null^forms("Romulus
Numbering").form("Starting Number") = Null^forms("Romulus
Numbering").form("Ending Number") = Null^forms("Romulus Numbering").form("From
Extension") = Null^forms("Romulus Numbering").form("To Extension") =
Null^DoCmd GoToControl "Prefix"^Rom_AddNumbers = True^AddNums_Exit:^DoCmd
Hourglass False^Close^Exit Function^AddNums_Err:^MsgBox "Error: " + Error$,
ROM_MB_OK + ROM_MB_ICONEXCLAMATION, ROMULUS_APP_NAME^Resume AddNums_Exit^End
Function^Function Rom_CloseNums ()^On Error GoTo CloseNums_Err^Dim ExtraText
As String^Dim ret As Integer^If forms("Romulus Numbering").form("Generate
Records") = True Then^ExtraText = " and use them to generate database
records"^Else^ExtraText = ""^End If^If gRom_NumsDirty = True Then^ret =
MsgBox("You have created a list of numbers. Do you want to print them" &
ExtraText & "?", ROM_MB_ICONEXCLAMATION + 3, ROMULUS_APP_NAME)^If ret = 6 Then
'^ Yes selected^ret = Rom_PrintNums()^GoTo CloseNums_Exit^End If^If ret = 2
Then '^ Cancel selected^GoTo CloseNums_Exit^End If^End If '^ No selected it

```

- 55 -

```

drops out^DoCmd Close A_FORM, "Romulus Numbering"^CloseNums_Exit:^Exit
Function^CloseNums_Err:^MsgBox "Error: " + Error$, ROM_MB_OK +
ROM_MB_ICONEXCLAMATION, ROMULUS_APP_NAME^Resume CloseNums_Exit^End
Function^Function Rom_OpenNumbering ()^On Error GoTo OpenNum_Err^DoCmd
OpenForm "Romulus Numbering"^OpenNum_Exit:^Exit Function^OpenNum_Err:^MsgBox
"Error: " + Error$, ROM_MB_OK + ROM_MB_ICONEXCLAMATION,
ROMULUS_APP_NAME^Resume OpenNum_Exit^End Function^Function Rom_PrintNums ()^On
Error GoTo PrintNums_Err^Dim ret As Integer^Dim FileName As String^Dim DB As
Database^Dim MainDyna As Dynaset^Dim DocNum As String^' * No longer needed - I
disable the buttons instead^' If gRom_NumsDirty = False Then^' MsgBox "You
have not created any numbers yet", ROM_MB_OK + ROM_MB_ICONEXCLAMATION,
ROMULUS_APP_NAME^' GoTo PrintNums_Exit^' End If^If Not IsNull(forms("Romulus
Numbering").form("Starting Number")) Then^ret = Rom_AddNumbers()^If ret = 0
Then GoTo PrintNums_Exit^End If^' * create the database records if
requested^If forms("Romulus Numbering").form("Generate Records") = True
Then^FileName = Rom_ReturnIniString(ROM_NUMFILE_INI_SECTION,
ROM_NUMFILE_INI_ENTRY)^Open FileName For Input As #1^Set DB = CurrentDB()^Set
MainDyna = DB.CreateDynaset("Romulus Database")^BeginTrans^On Error Resume
Next^Input #1, DocNum^While Err = 0^MainDyna.AddNew^MainDyna("Doc Number") =
DocNum^MainDyna.Update^Err = 0 ^' * so that if there is a duplicate key error we
keep going^Input #1, DocNum^Wend^CommitTrans^DB.Close^MainDyna.Close^On Error
GoTo PrintNums_Err^Close^End If^' * according to selection set Avery labels up
to open correct file^If forms("Romulus Numbering").form("Bar Codes") = True
Then^FileName = Rom_ReturnIniString(ROM_NUMFILE_INI_SECTION,
ROM_BARFILE_INI_ENTRY)^Else^FileName =
Rom_ReturnIniString(ROM_NUMFILE_INI_SECTION, ROM_PLAINFILE_INI_ENTRY)^End
If^ret = Rom_WritePrivateProfileString(ROM_AVERY_LAYOUT_SECTION,
ROM_AVERY_LASTFILE_ENTRY, FileName, ROM_AVERY_INI_NAME)^' * open Avery Labels
or make it the active window^ret = Rom_OpenApp(ROM_LABEL_INI_ENTRY,
ROM_AVERY_WINDOWCLASS, ROM_AVERY_WINDOWNAME)^'^' * update display of the
linked item - needed if already open^' * open the filename^' * ??? problem
getting the list to link on first opening - so will rely on a label that is
already linked opening automatically^' FileName =
Rom_ReturnIniString(ROM_NUMFILE_INI_SECTION, ROM_NUMFILE_INI_ENTRY)^'If ret =
True Then^'SendKeys "%ml" & FileName &
"{TAB}{TAB}{TAB}{DOWN}{UP}{UP}{UP}{UP}{UP}{UP}{UP}{DOWN}{ENTER}{ENTER}%ml{
ENTER}"^'Else^'GoTo PrintNums_Exit^'End If^'^DoCmd Close A_FORM, "Romulus
Numbering"^PrintNums_Exit:^Close^Exit Function^PrintNums_Err:^MsgBox "Error: "

```

- 56 -

```

+ Error$, ROM_MB_OK + ROM_MB_ICONEXCLAMATION, ROMULUS_APP_NAME^Resume
PrintNums_Exit^End Function^Function Rom_SetupNumbering ()^On Error GoTo
SetupNum_Err^^ * set module variable to indicate we are now
clean^gRom_NumsDirty = False^Dim FileName As String^^ * set up the numbering
file as empty^FileName = Rom_ReturnIniString(ROM_NUMFILE_INI_SECTION,
ROM_NUMFILE_INI_ENTRY)^Open FileName For Output As
#1^SetupNum_Exit:^Close^Exit Function^SetupNum_Err:^MsgBox "Error: " + Error$,
ROM_MB_OK + ROM_MB_ICONEXCLAMATION, ROMULUS_APP_NAME^Resume SetupNum_Exit^End
FunctionFunction Rom_ViewNums ()^On Error GoTo ViewNum_Err^Dim ret As
Integer^Dim FileName As String^^ * No longer needed - I disable the buttons
instead^^ If gRom_NumsDirty = False Then^^ MsgBox "You have not created any
numbers yet", ROM_MB_OK, ROMULUS_APP_NAME^^ GoTo ViewNum_Exit^^ End
If^FileName = Rom_ReturnIniString(ROM_NUMFILE_INI_SECTION,
ROM_NUMFILE_INI_ENTRY)^ret = Shell("notepad " & FileName,
1)^ViewNum_Exit:^Exit Function^ViewNum_Err:^MsgBox "Error: " + Error$,
ROM_MB_OK + ROM_MB_ICONEXCLAMATION, ROMULUS_APP_NAME^Resume ViewNum_Exit^End
Function^^^

```

### Claims

1. A method of facilitating the use of a plurality of G.U.I. applications, the method comprising running a first G.U.I. application with a first data file or record loaded and at the same time displaying one or more primary symbols representing one or more further G.U.I. applications, and, on actuation of one of the primary symbols by a user, searching for a second data file or record associated with said first data file or record and, if such an associated second data file or record exists, causing the further G.U.I. application represented by said one of the primary symbols to be run with the associated second data file or record automatically loaded.

2. A method as claimed in Claim 1, wherein two data files or records are considered to be associated when their names are the same, or when designated parts, for example the first parts, of their names are the same.

3. A method as claimed in Claim 1 or 2, which further includes, while the first G.U.I. application is running, displaying one or more secondary symbols representing said one or more further G.U.I. applications, and, on actuation of one of said secondary symbols by a user, running the G.U.I. application associated with that secondary symbol with no data file or record automatically loaded.

4. A method as claimed in any preceding claim, which further includes the step of, when one of the further G.U.I. applications is running, displaying a return symbol representing the first G.U.I. application, and, on actuation of the return symbol by a user, running the first G.U.I. application.

5. A method as claimed in Claim 4, wherein the first G.U.I. application is run with a third data file or record automatically loaded if, when the return symbol is actuated, said one of the further G.U.I. applications is running with a fourth data file or record loaded and the third and fourth data files or records are associated with each other.

6. A method as claimed in any preceding claim, which further includes, when one of the further G.U.I. applications is running with a particular data file or record loaded, displaying one or more further application symbols representing the other further applications, and, on actuation of one of the further application symbols by a user, causing the further G.U.I. application represented by said one of the further application symbols to be run with a data file or record automatically loaded which is associated with said particular data file or record.

7. A method as claimed in any preceding claim, wherein the user is able to specify, at any stage during the method, the default directories to be used by the G.U.I. applications (including the first G.U.I. application) when the G.U.I. applications are caused to be run by actuation of a relevant symbol.

8. A method as claimed in Claim 7, wherein two data files or records can only be associated with each other, for example by virtue of their names, when each of the data files or records is located in a respective default directory.

9. A method as claimed in Claim 8, which further includes displaying a number generating symbol and, on actuation of the number generating symbol, and input of start and finish values, by the user, generating a sequence of numbers, or alphanumeric strings, between the start and finish values.

10. A method as claimed in Claim 9, which also includes generating a bar code corresponding to each number or string, and/or printing a label corresponding to each number or string, and/or creating a data file or record having a name corresponding to each string.

11. A device for carrying out the method claimed in any one of Claims 1 to 10, the device comprising computing means for running the G.U.I. applications, display means for displaying said symbols and information relating to the G.U.I. applications, first input means for allowing the user to operate the G.U.I. applications, and second input means for allowing the user to actuate said symbols.

12. A method of allocating virtual memory in a GUI environment, said method comprising the steps of:

providing an allocation request for virtual memory;  
increasing said allocation request to define an extra virtual memory space for storing GUI objects; and  
allocating an increased virtual memory space in response to said increased allocation request when said increased allocation request is less than or equal to the amount of free virtual memory available.

13. The method according to claim 12 further comprising the step of writing overflow data in said allocated increased virtual memory space.

14. The method according to claims 11 or 13 wherein said allocation request for virtual memory is provided for an application.

15. The method according to claim 14 further comprising the step of releasing said allocated virtual memory when said application is closed.

16. The method according to any one of claims 12-15 wherein said allocation request is increased by a percentage value of said virtual memory.

17. The method according to any one of claims 12-16 wherein said GUI objects comprise a window name and a window class.

18. The method according to claim 17 wherein said GUI objects further comprise a desk class, or a child class, or a child class and a desk class.

19. The method according to any one of claims 12-18 wherein said GUI environment is Microsoft Windows®.

20. A method of allocating windows resources in the Microsoft Windows<sup>®</sup> environment, said method comprising the steps of:

providing an allocation request for windows resources;

increasing said allocation request to define a buffer for storing GUI objects;

5 and

allocating an increased portion of said windows resources in response to said increased allocation request when said increased allocation request is less than or equal to the amount of free windows resources available.

21. The method according to claim 20 further comprising the step of  
10 storing overflow data in said allocated increased portion of windows resources.

22. The method according to claims 20 or 21 wherein said allocation request for windows resources is provided for an application.

23. The method according to any one of claims 20-22 wherein said allocation request is increased by a percentage value of said windows resources.

15 24. The method according to any one of claims 20-23 wherein said GUI objects comprise a window name and a window class.

25. The method according to claim 24 wherein said GUI objects further comprise a desk class, or a child class, or a desk class and a child class.

26. A method of managing windows resources in the Microsoft Windows<sup>®</sup>  
20 environment, said method comprising the steps of:

allocating an increased percentage of windows resources for an application displayed in a main window, said increased percentage of windows resources comprising a nominal portion of windows resources for said application and a buffer which are contiguously arranged in relation to one another; and

25 storing windows resources of one or more child windows of said main window in said buffer.

27. The method according to claim 26 wherein said step of allocating an increased percentage of windows resources comprises the steps of:

30 obtaining a request for said nominal portion of said windows resources generated by opening said application in said main window;

increasing said request by a factor for allocating said buffer contiguously with said nominal portion of said windows resources;

35 reserving said increased percentage of windows resources in response to said increased request for windows resources when said increased request is less than or equal to the amount of free windows resources available.

28. The method according to claim 27 wherein said step of storing windows resources of one or more child windows of said main window in said buffer comprises the steps of:

obtaining a second request for windows resources generated by a child window; and

providing a portion of said buffer for said child window in response to said second request when said second request is less than or equal to the amount of free  
5 windows resources available in said buffer.

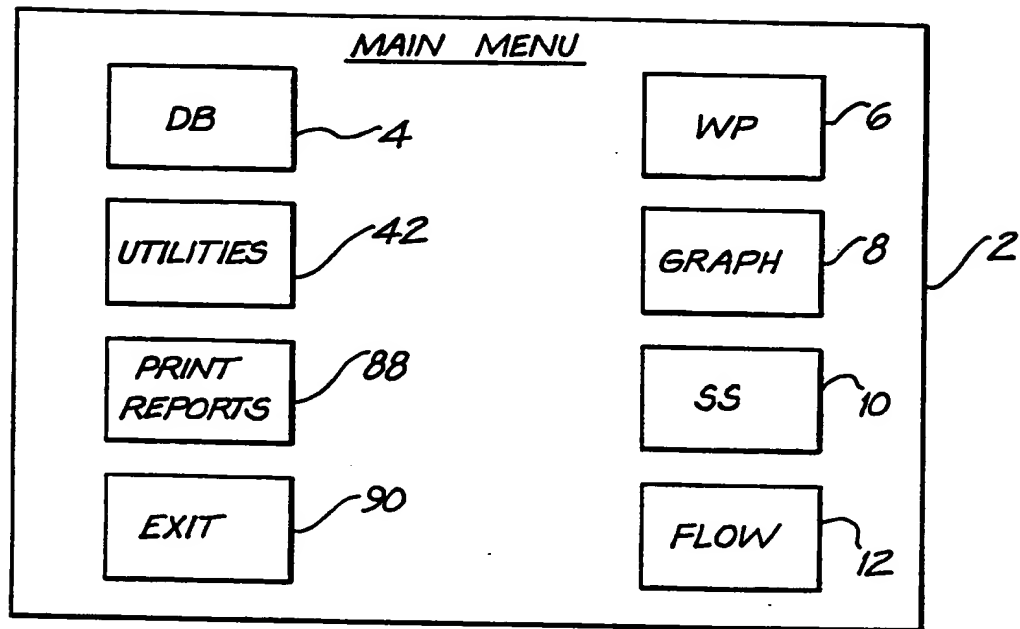


FIG. 1

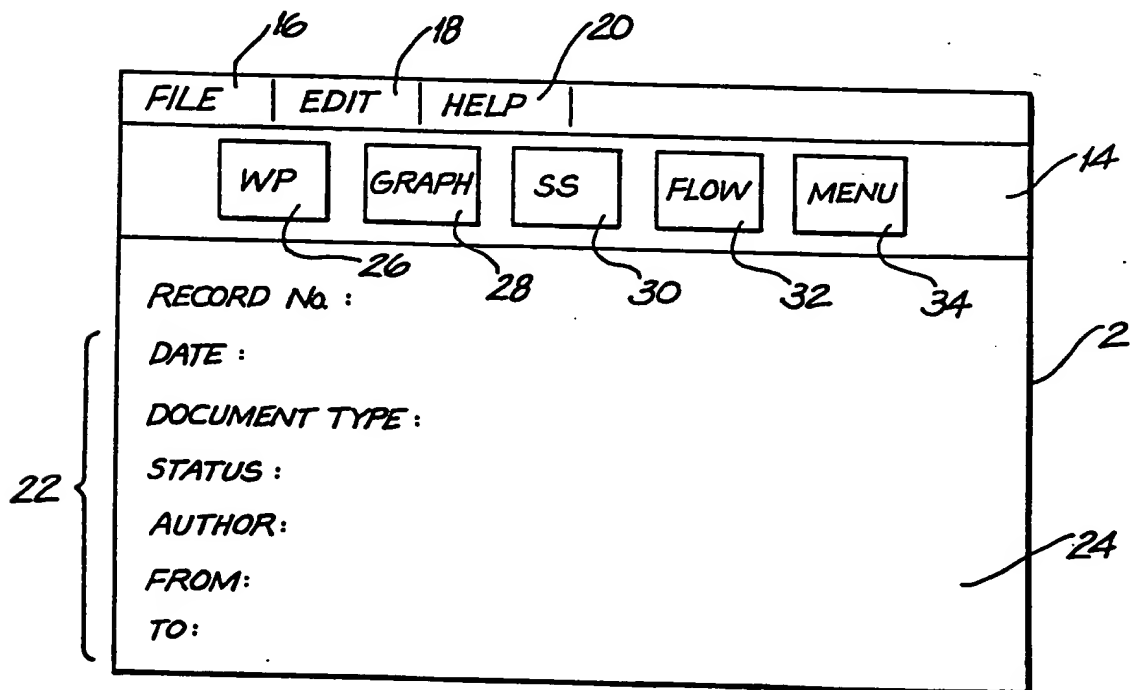


FIG. 2

2/9

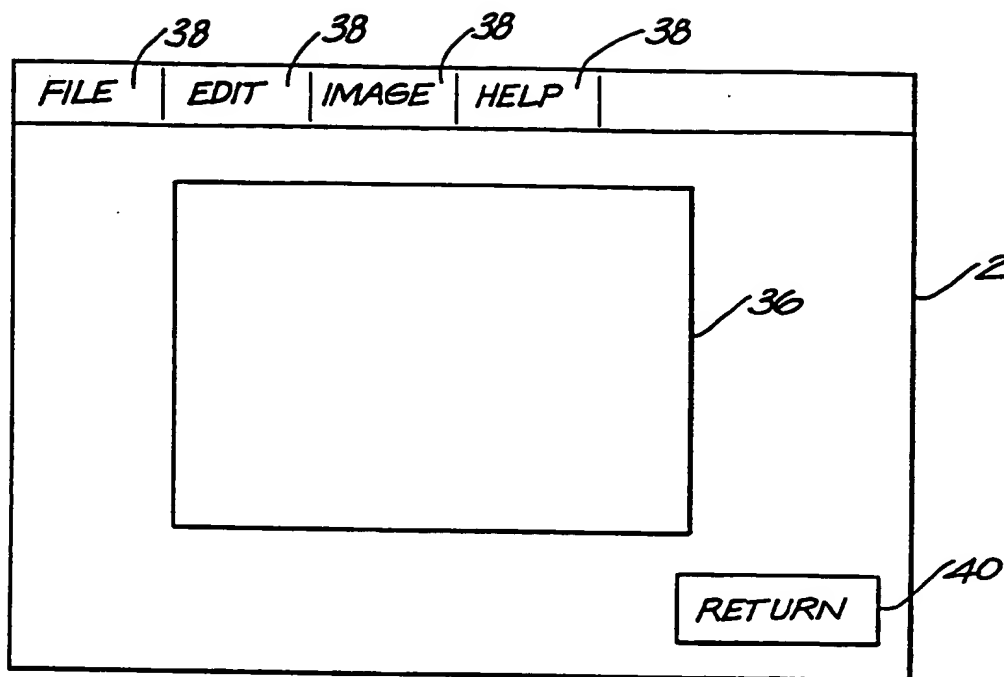


FIG. 3

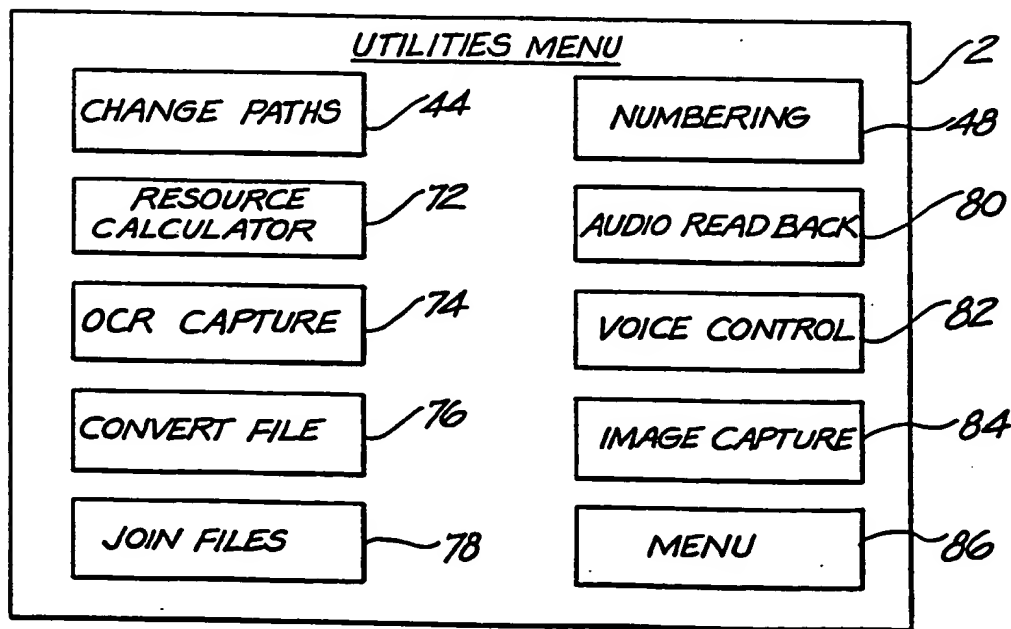


FIG. 4

PATHS

WP	<input type="text"/>	46
GRAPH	<input type="text"/>	46
SS	<input type="text"/>	46
FLOW	<input type="text"/>	46

2

FIG. 5

NUMBERING

PREFIX	<input type="text"/>	56	<input type="button" value="GENERATE"/>	54
STARTING No.	<input type="text"/>	50	<input type="button" value="VIEW"/>	66
ENDING No.	<input type="text"/>	52	<input type="button" value="PRINT"/>	68

ADD EXTENSIONS

FROM:	<input type="text"/>	58
TO:	<input type="text"/>	60

62 ☐ BAR CODES

64 ☐ DATABASE RECORDS

70

2

FIG. 6

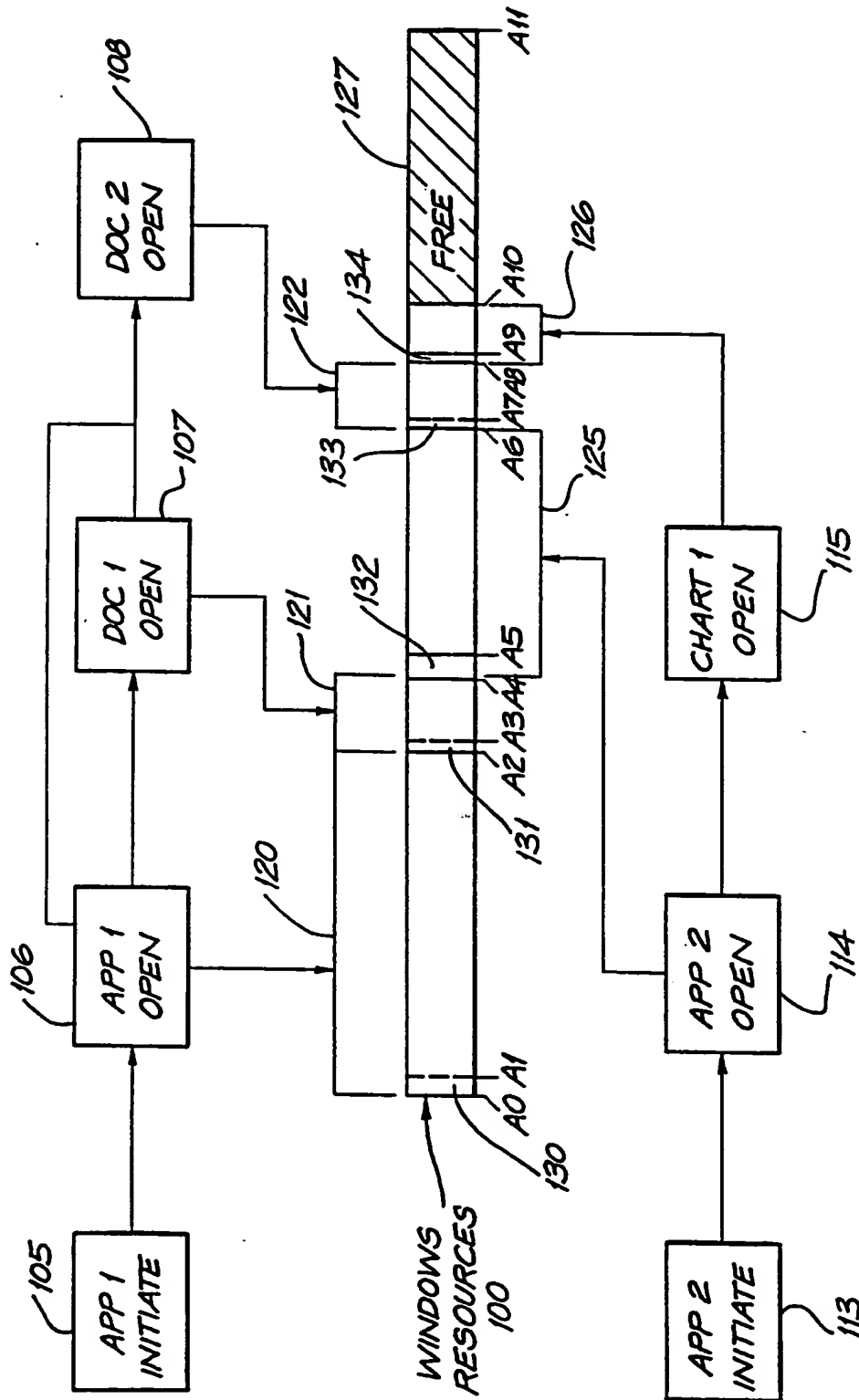


FIG. 7A  
(PRIOR ART)

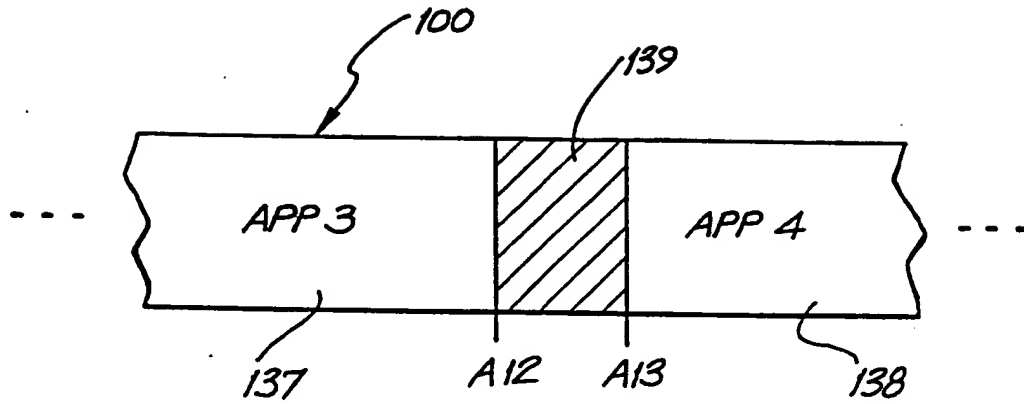


FIG. 7B

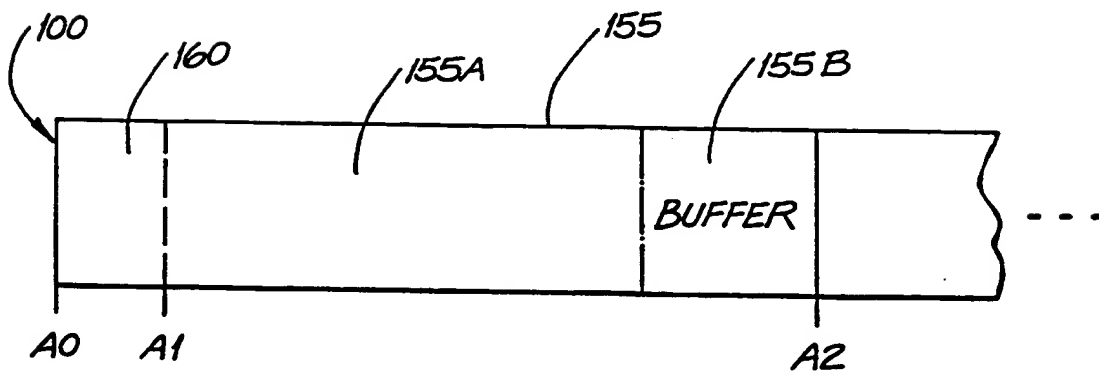


FIG. 8B

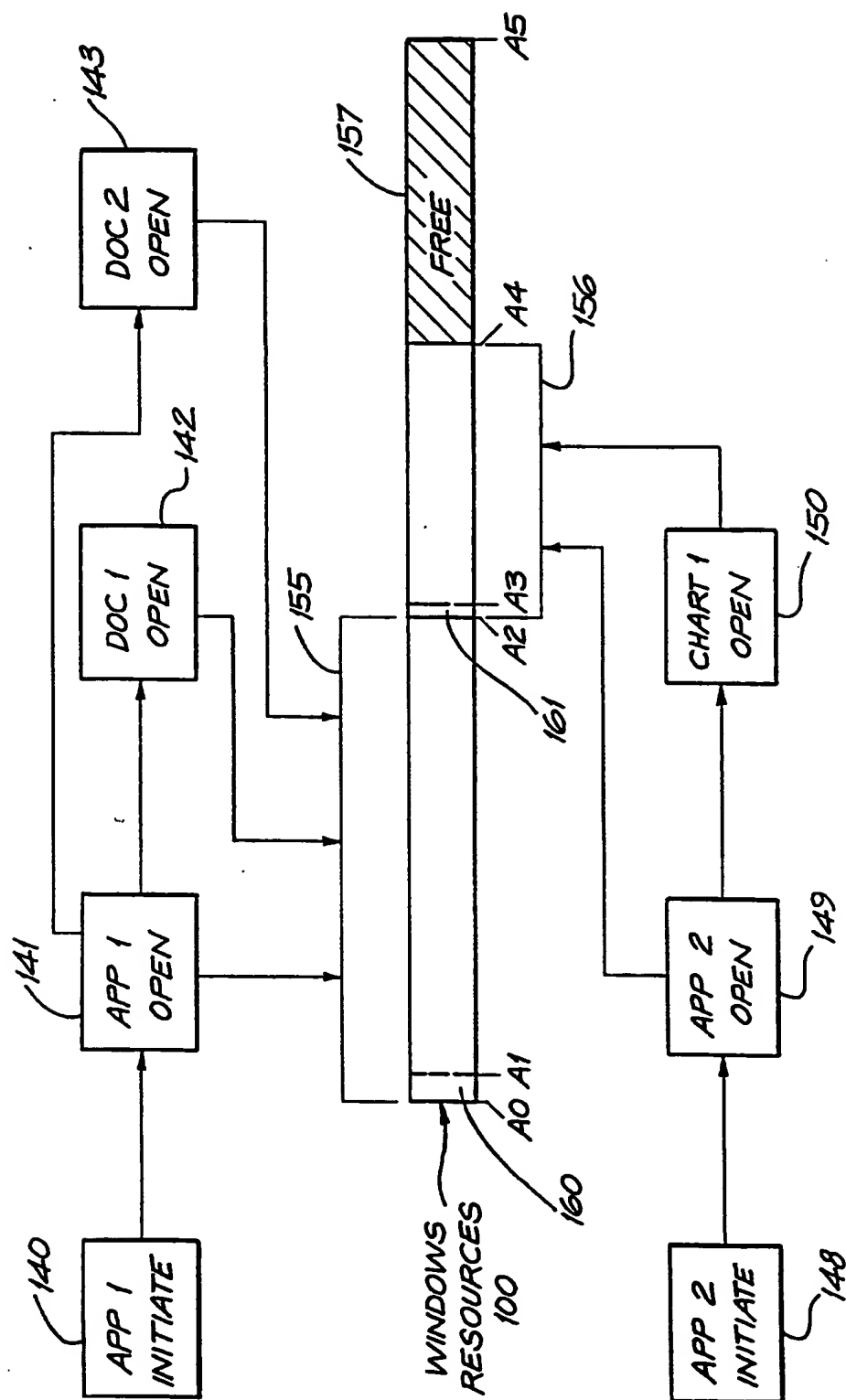
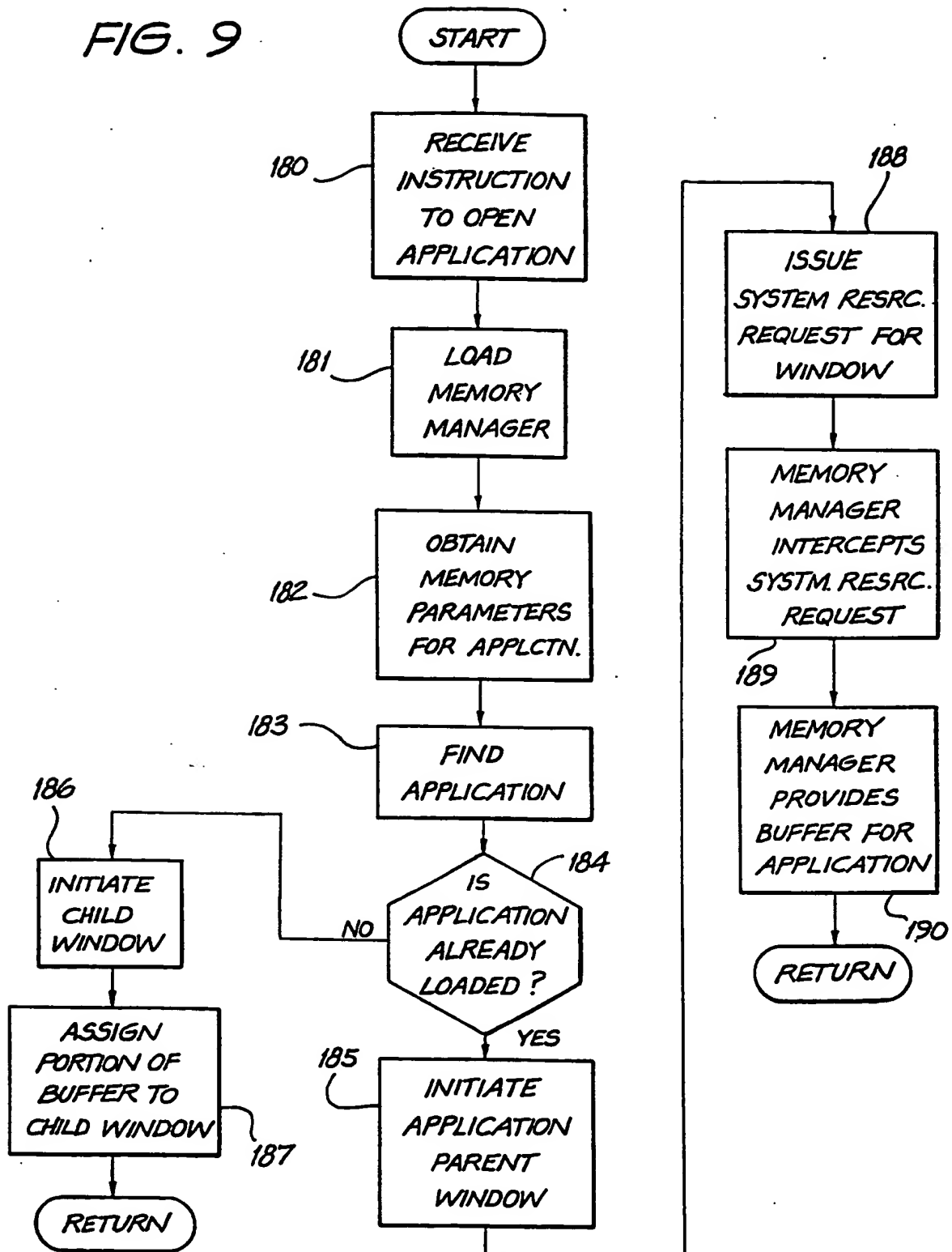


FIG. 8A

7/9

FIG. 9



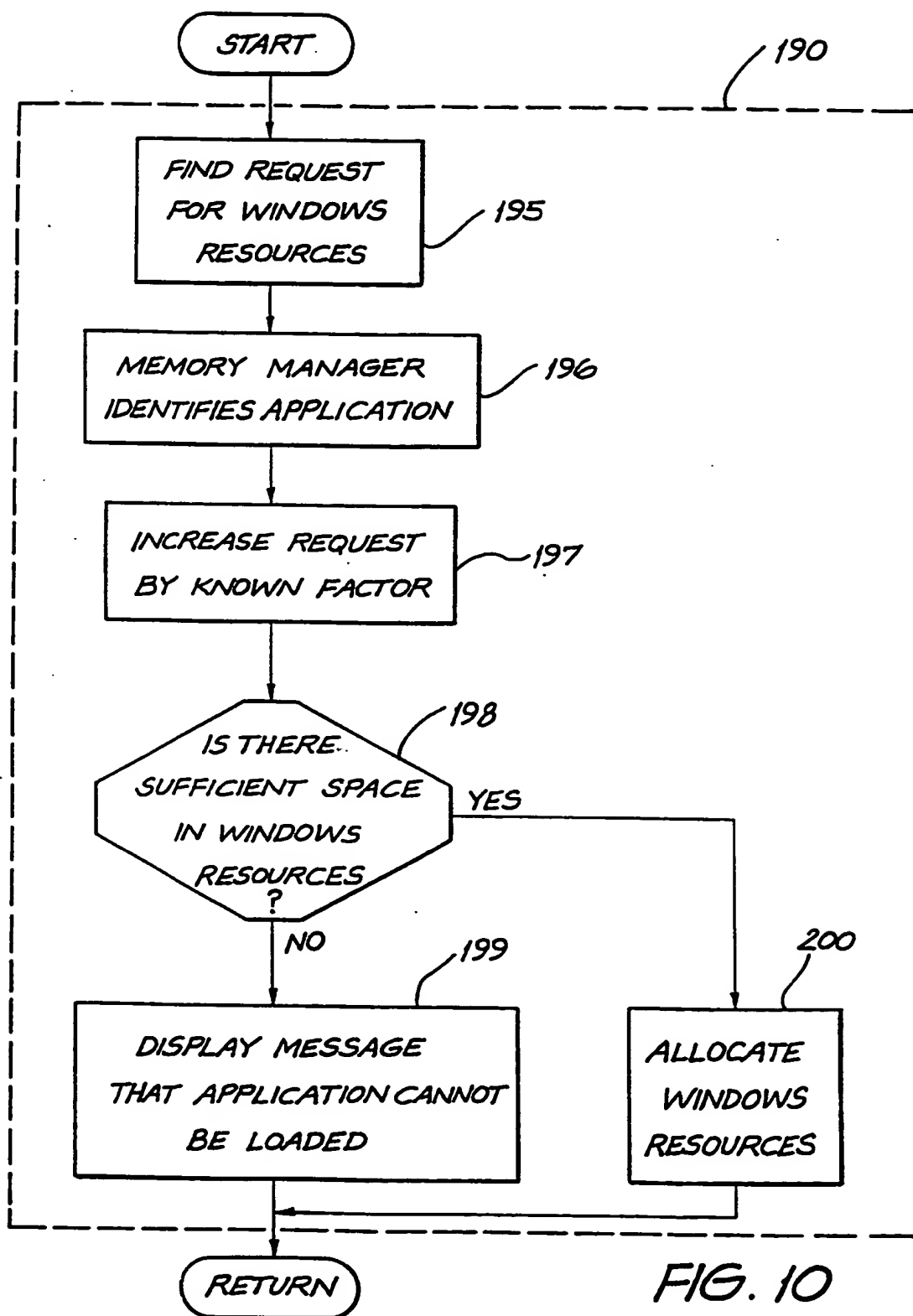


FIG. 10

9/9

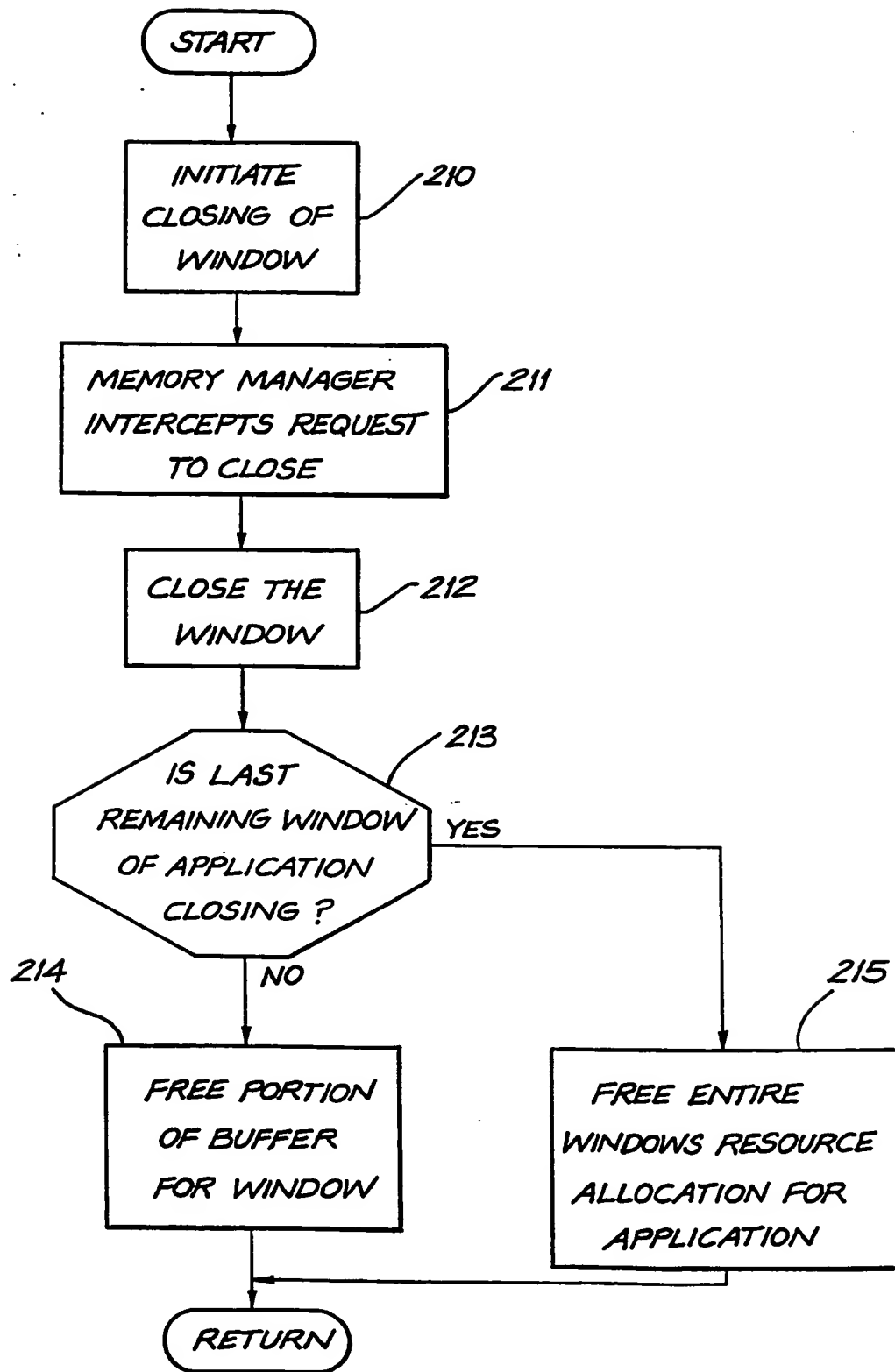



FIG. 11

# INTERNATIONAL SEARCH REPORT

International application No.

PCT/AU 95/00068

<b>A. CLASSIFICATION OF SUBJECT MATTER</b> Int. Cl. <sup>6</sup> G06F 9/445, 12/08, 12/02  According to International Patent Classification (IPC) or to both national classification and IPC					
<b>B. FIELDS SEARCHED</b>  Minimum documentation searched (classification system followed by classification symbols) IPC G06F 9/445, 9/44, 9/46, 12/02, 12/08  Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched AU: IPC as above  Electronic data base consulted during the international search (name of data base, and where practicable, search terms used)					
<b>C. DOCUMENTS CONSIDERED TO BE RELEVANT</b>					
Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to Claim No.			
X	EP, A2, 456275 (FUJITSU LTD) 13 November 1991 Column 2, lines 24-30	12-28			
X	Andrew S. Tanenbaum "Operating Systems: Design and Implementation", published 1987 by prentice-Hall International, Englewood Cliffs (New Jersey) paragraph bridging pages 199-200	12-28			
A	Brian Livingston "Windows 3 Secrets", published 1991 by IDG Books Worldwide, San Mateo (California), pages 100-105	1-11			
<div style="display: flex; justify-content: space-between;"> <div> <input checked="" type="checkbox"/> Further documents are listed in the continuation of Box C.         </div> <div> <input checked="" type="checkbox"/> See patent family annex.         </div> </div>					
<table style="width: 100%; border: none;"> <tr> <td style="width: 33%; vertical-align: top;">           * Special categories of cited documents :            "A" document defining the general state of the art which is not considered to be of particular relevance            "E" earlier document but published on or after the international filing date            "L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)            "O" document referring to an oral disclosure, use, exhibition or other means            "P" document published prior to the international filing date but later than the priority date claimed         </td> <td style="width: 33%; vertical-align: top;">           "T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention            "X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone            "Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art            "&amp;" document member of the same patent family         </td> <td style="width: 33%;"></td> </tr> </table>			* Special categories of cited documents : "A" document defining the general state of the art which is not considered to be of particular relevance "E" earlier document but published on or after the international filing date "L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified) "O" document referring to an oral disclosure, use, exhibition or other means "P" document published prior to the international filing date but later than the priority date claimed	"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention "X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone "Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art "&" document member of the same patent family	
* Special categories of cited documents : "A" document defining the general state of the art which is not considered to be of particular relevance "E" earlier document but published on or after the international filing date "L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified) "O" document referring to an oral disclosure, use, exhibition or other means "P" document published prior to the international filing date but later than the priority date claimed	"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention "X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone "Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art "&" document member of the same patent family				
Date of the actual completion of the international search 29 May 1995	Date of mailing of the international search report 2 JUNE 1995 (02.06.95)				
Name and mailing address of the ISA/AU  AUSTRALIAN INDUSTRIAL PROPERTY ORGANISATION PO BOX 200 WODEN ACT 2606 AUSTRALIA  Facsimile No. 06 2853929	Authorized officer  <div style="text-align: center; margin-top: 20px;">   <b>ALLEN EVANS</b> </div> Telephone No. (06) 2832209				

**INTERNATIONAL SEARCH REPORT**

International application No.

**PCT/AU 95/00068**

C(Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT		
Category*	Citation of document, with indication, where appropriate of the relevant passages	Relevant to Claim No.
A	"Microsoft Windows User's Guide for the Windows Graphical Environment", published 1985 by Microsoft Corporation pages 135-137	1-11
A	EP,A2, 325885 (INTERNATIONAL BUSINESS MACHINES CORPORATION) 2 August 1989, see particularly Abstract	

**INTERNATIONAL SEARCH REPORT**

International application No.

**PCT/AU 95/00068****Box I Observations where certain claims were found unsearchable (Continuation of Item 1 of first sheet)**

This international search report has not established in respect of certain claims under Article 17(2)(a) for the following reasons:

1. ☐ Claims Nos.:  
because they relate to subject matter not required to be searched by this Authority, namely:
  
2. ☐ Claim Nos.:  
because they relate to parts of the international application that do not comply with the prescribed requirements to such an extent that no meaningful international search can be carried out, specifically:
  
3. ☐ Claims Nos.:  
because they are dependent claims and are not drafted in accordance with the second and third sentences of Rule 6.4(a).

**Box II Observations where unity of invention is lacking (Continuation of item 2 of first sheet)**

This International Searching Authority found multiple inventions in this international application, as follows:

1. Claims 1-11

A method or device for facilitating the use of a plurality of GUI applications,

2. Claims 2-28

A method of allocating memory or other resources in a Microsoft Windows or other GUI environment as reasoned on the extra sheet:

1. ☒ As all required additional search fees were timely paid by the applicant, this international search report covers all searchable claims
2. ☐ As all searchable claims could be searched without effort justifying an additional fee, this Authority did not invite payment of any additional fee.
3. ☐ As only some of the required additional search fees were timely paid by the applicant, this international search report covers only those claims for which fees were paid, specifically claims Nos.:
  
4. ☐ No required additional search fees were timely paid by the applicant. Consequently, this international search report is restricted to the invention first mentioned in the claims; it is covered by claims Nos.:

**Remark on Protest**

- ☐ The additional search fees were accompanied by the applicant's protest.
- ☒ No protest accompanied the payment of additional search fees.

(continuation) of Box II

The International Application does not comply with the requirements of unity of invention because it does not relate to one invention or to a group of inventions so linked as to form a single inventive concept. In coming to this conclusion the international Searching Authority has found that there are two inventions.

1. Claims 1-11 are directed to a method or device for facilitating the use of a plurality of GUI applications. The "special technical feature" of these claims is the running of a further GUI application with a data file automatically loaded, where that data file is associated with the data file of GUI application which is already running.

2. Claims 12-28 are variously directed to "a method of allocating virtual memory in a GUI environment", "a method of allocating windows managing windows resources in the Microsoft Windows environment". The special technical feature of these claims is allocating an amount of memory or resources which is increased over the amount requested or otherwise required.

Since the above-mentioned groups of claims do not share either of the technical features identified, a "technical relationship" between the inventions, as defined in PCT Rule 13.2, does not exist. Accordingly the international Application does not relate to one invention or to a single inventive concept.

**PCT/AU 95/00068**

[illegible]